

**Oracle® Communications Instant Messaging
Server**

System Administrator's Guide

Release 10.0.1

E76291-01

August 2016

Copyright © 2015, 2016, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface	xiii
Audience.....	xiii
Related Documents	xiii
Documentation Accessibility	xiii
 Part I Monitoring and Managing Instant Messaging Server	
 1 Instant Messaging Server Administration Overview	
Overview of Instant Messaging Server Administration Tasks.....	1-1
About Instant Messaging Server Commands	1-1
Directory Placeholders Used in This Guide	1-2
 2 Administering Instant Messaging Server Components	
Overview of Stopping, Starting, Refreshing, and Checking Instant Messaging Server Components.....	2-1
Using Service Management Framework (SMF).....	2-1
Starting Instant Messaging Server Components	2-2
Starting All Components	2-2
Starting a Single Component.....	2-2
Stopping Instant Messaging Server Components	2-2
Stopping All Components	2-3
Stopping a Single Component	2-3
Registering and Unregistering Service Management Framework.....	2-3
Registering SMF	2-3
Unregistering SMF.....	2-3
Refreshing Component Configuration	2-3
Refreshing All Components	2-4
Refreshing a Single Component	2-4
Checking the Status of Instant Messaging Server Components.....	2-4
Changing Instant Messaging Server and Multiplexor Configuration Properties.....	2-5
Changing Configuration Properties	2-5
Backing Up Instant Messaging Server Data	2-5
Backup Information	2-5
Performing a Backup	2-5
Restoring Backup Information	2-6

3 Using LDAP with Instant Messaging Server

Managing LDAP Access Configuration for Instant Messaging Server	3-1
Overview of How Instant Messaging Server Uses LDAP	3-1
Searching the Directory Anonymously	3-1
Enabling the Server to Conduct Directory Searches as a Specific End User	3-2
Using the mail Attribute for User Authentication	3-2
Using LDAP Groups	3-3
Configuring Instant Messaging Server to Use LDAP Groups	3-3
Using Group Messaging	3-4
Configuring Chat Room Membership Based on LDAP Group Membership	3-4
Using Chat Rooms Based on LDAP Group Membership	3-5
Converting Associated Domain to Canonical Domain	3-5

4 Using Oracle Database with Instant Messaging Server

Overview of Using Oracle Database for Storing Messages	4-1
Installing and Creating an Oracle Database Instance for Instant Messaging Server	4-1

5 Administering Instant Messaging Server End Users

Overview of Administering Instant Messaging Server End Users	5-1
Registering New Instant Messaging Server Users	5-1
Configuring Instant Messaging Server to Enable New User Registration	5-1
Allowing New User Registration Using Policy Module iim_ldap	5-2
Allowing New User Registration Using Policy Module schema1 or schema2	5-2
Storing Instant Messaging Server User Properties in LDAP	5-2

6 Managing Messages for Users

About Managing Instant Messaging Server Messages for Offline Users	6-1
About Delivering Offline Messages	6-1
Enabling Offline Messaging	6-2
About Push Notifications for Offline Messages	6-3
Enabling Push Notifications for Offline Messages	6-4
About Message Carbons	6-4
Enabling Message Carbons	6-4

7 Improving Instant Messaging Server Performance

Tuning Instant Messaging Server Memory	7-1
Tuning Java Virtual Machine Options	7-1
Instant Messaging Server and Multiplexor Thread Pooling and Service Port Configuration ..	7-2
Creating Service Port Configurations	7-3
Creating Multiplexor Thread Pools	7-3
Sample Load Test of the Instant Messaging Server	7-4
Controlling the Rate of Incoming Connections	7-5
Tuning the XMPP WebSocket Gateway	7-6
Configuring Multiplexor Thread Pools	7-6

8 Configuring Instant Messaging Server for High Availability

Overview of High Availability for Instant Messaging Server	8-1
About Server Pooling	8-1
Availability in an Instant Messaging Server Pool	8-2
Configuring Server-to-Server Communication Between Instant Messaging Servers.....	8-2
Setting Up Communication Between Two Instant Messaging Servers in a Server Pool.....	8-3
Adding a New Node to an Existing Instant Messaging Server Deployment	8-3
Securing a Multi-node Deployment.....	8-4
Manually Defining the Dialback Key for an Instant Messaging Server in a Server Pool	8-4
Using Shoal for Server Pool Messaging	8-4
Setting Shoal Properties	8-4
Using Shoal for Automatic Discovery of Peer Servers in a Pool.....	8-5
Using Shoal for Conferences Across Server Pools	8-5
Using Shoal Across Subnets	8-6
About Multiplexor Failover	8-6
Enabling Multiplexor Failover	8-7
Overview of Using Oracle Solaris Cluster	8-7
HA Configuration Software Requirements.....	8-7
HA Configuration Requirements.....	8-8
HA Configuration Terms and Checklist.....	8-8
Starting and Stopping the Instant Messaging Server HA Service	8-9
Troubleshooting the Instant Messaging Server HA Configuration.....	8-9
Setting Up HA for Instant Messaging Server	8-9
Choosing a High Availability Model for Your Instant Messaging Server Deployment.....	8-9
High-Level Task List for an Asymmetric HA Deployment.....	8-10
High-Level Task List for a Symmetric HA Deployment.....	8-10
Installing and Configuring in an Asymmetric HA Environment.....	8-11
Installing and Configuring in a Symmetric HA Environment.....	8-15
Removing HA for Instant Messaging Server.....	8-22

9 Configuring LDAP Failover

Overview of Configuring LDAP Failover	9-1
Setting Up LDAP Failover.....	9-1

10 Managing Archiving in Instant Messaging Server

About Archiving.....	10-1
Enabling and Disabling Archiving for Instant Messaging Server.....	10-1
Enabling Instant Messaging Server Archiving	10-2
Disabling Instant Messaging Server Archiving	10-2
Archiving in Instant Messaging Server	10-2
Managing Instant Messaging Server File Archive	10-2
Implementing the Custom File Architecture Provider.....	10-2
File Archiver Provider Example	10-3
Compiling the Custom File Archival Provider Application.....	10-4
Enabling and Disabling the Instant Messaging Server File Archive Provider.....	10-4
Enabling File Archiving	10-4

Disabling File Archiving	10-4
Managing Instant Messaging Server Message Archive.....	10-4
Implementing the Custom Message Archival Provider	10-4
Message Archive Provider Example	10-7
Compiling the Custom Message Archival Provider Application.....	10-8
Enabling and Disabling the Instant Messaging Server Message Archive Provider.....	10-8
To Disable Message Archiving.....	10-9
Managing Instant Messaging Server Email Archive.....	10-9
Enabling and Disabling the Instant Messaging Server Email Archive Provider.....	10-9
Enabling the Instant Messaging Server Email Archive.....	10-9
Disabling the Instant Messaging Server Email Archive Provider	10-9
Configuring Email Archive Settings	10-9
Configuring Administrator Recipients and the RFC 822 Header Format.....	10-10
Email Header Format	10-10
RFC 822 Email Archive Header Fields for One to One Chat.....	10-10
RFC 822 Email Archive Header Fields for Private Conferences	10-11
RFC 822 Email Archive Header Fields for Public Conferences	10-11
RFC 822 Email Archive Header Fields for Poll Questions with Replies.....	10-11
RFC 822 Email Archive Header Fields for Poll Replies Only.....	10-11
RFC 822 Email Archive Header Fields for Alerts.....	10-11
RFC 822 Email Archive Header Fields for New Channel Posts.....	10-12
Enabling and Disabling the Instant Messaging Server Custom Archive Provider	10-12
Enabling a Custom Archive Provider	10-12
Disabling a Custom Archive Provider	10-12
About Archived Messages Retrieval	10-13
Archived Message Management Example	10-13
Enabling Retrieval of Archived Messages.....	10-13
Message Archive Retrieval Provider Example	10-13
Using Chat Markers	10-14

11 Managing Message Conversion in Instant Messaging

About Message Conversion	11-1
Managing Message Conversion in the Instant Messaging Server	11-1
Implementing the Custom Message Conversion Provider.....	11-1
Message Converter Provider Example	11-2
Compiling the Custom Message Converter Provider	11-2
Enabling and Disabling the Instant Messaging Message Converter Provider.....	11-3

12 Monitoring Instant Messaging Server and Multiplexor

Overview of Monitoring the Instant Messaging Server.....	12-1
Configuring Instant Messaging Server Monitoring.....	12-1
Steps for Configuring Data Collection and Monitoring.....	12-2
Installing and Configuring the Oracle Enterprise Manager Plug-in	12-3
Product Version Requirements	12-4
Installation Prerequisites.....	12-4
Downloading the Enterprise Manager Plug-in	12-4
Configuring Instant Messaging Server Targets	12-4

Adding Instant Messaging Server Host Targets and Installing the Management Agent	12-4
Setting Up Preferred Credentials	12-5
Deploying the Enterprise Manager Cloud Control Plug-in.....	12-5
Deploying the Enterprise Manager Cloud Control Plug-in on the Management Server	12-6
Deploying the Enterprise Manager Cloud Control Plug-in on Host Targets	12-6
Using the Enterprise Manager to Monitor Instant Messaging Server	12-6
Viewing Metrics	12-7
Enabling and Using the Beacon Service	12-7
Available Server Metrics	12-8
Customizing Monitoring	12-11
Setting Thresholds on Monitored Metrics	12-11
Setting Notification Options	12-11
Adding Corrective Actions	12-11
Instant Messaging Server Diagnostic Metrics	12-11
Enabling and Disabling Instant Messaging Server Diagnostic Metrics	12-12
Overview of Monitoring the Instant Messaging Multiplexor.....	12-12
Configuring Instant Messaging Multiplexor Monitoring	12-13
Steps for Configuring Data Collection and Monitoring	12-13
Available Multiplexor Metrics	12-15
Instant Messaging Multiplexor Diagnostic Metrics.....	12-15
Enabling and Disabling Instant Messaging Server Diagnostic Metrics	12-15

13 Troubleshooting Instant Messaging Server

Troubleshooting and Monitoring Instant Messaging Server Overview.....	13-1
General Troubleshooting.....	13-1
Cannot Forward Mail to Offline Users	13-1
Configuring the Attribute Used for User Email Addresses	13-1
Calendar Pop-up Reminders Do Not Work.....	13-1
Connection Refused or Timed Out.....	13-2
Authentication Errors	13-2
Instant Messaging Server Content is not Archived.....	13-2
Server-to-Server Communication Fails to Start	13-2
Troubleshooting Instant Messaging Server and LDAP.....	13-3
Using a Directory That Does not Permit Anonymous Bind	13-3
Configuring Bind Credentials for Instant Messaging Server	13-3
Changing the Attribute Used to Display Contact Names.....	13-3
Searching the Directory by Using Wildcards.....	13-3
Using Nonstandard Objectclasses for Users and Groups	13-3
Changing the Object Classes Used to Specify Users and Groups.....	13-4
Using an Attribute Other than uid for User Authentication	13-4
Changing the Attribute Used for User Authentication	13-4
Using an Attribute Other than uid for User IDs.....	13-4
Changing the Attribute Used for User IDs.....	13-4
Troubleshooting Connectivity Issues in a Multi-Node Deployment (Server Pool)	13-5
Managing the Watchdog Process	13-5
Determining the Status of the Watchdog	13-5
Enabling and Disabling the Watchdog	13-5

Managing Logging for the Watchdog.....	13-6
14 Managing Logging for Instant Messaging Server	
Instant Messaging Server Log File Location.....	14-1
Instant Messaging Server Component Logging Levels.....	14-1
Managing Instant Messaging Server Logging by Using log4j.....	14-2
Instant Messaging Server Log4j Configuration File (log4j.conf) Location.....	14-2
Instant Messaging Server Log4j Log File Syntax.....	14-2
Log4j Log Levels for Instant Messaging Server Components.....	14-3
Specifying the Location of the log4j Configuration File (log4j.conf).....	14-3
Enabling or Disabling log4j Logging for a Component.....	14-4
Setting log4j Log Levels.....	14-4
Specifying the Maximum log4j Log File Size.....	14-4
Part II Configuring Gateways, Protocols, and Features	
15 Configuring Hosted Domain Support	
About Instant Messaging Server Hosted Domains.....	15-1
Setting Up Schema 1 and Schema 2 for Instant Messaging Server Hosted Domains.....	15-1
Schema 1 Structure.....	15-1
Configuring Instant Messaging Server for Schema 1.....	15-2
Schema 2 Structure.....	15-2
Configuring Instant Messaging Server for Schema 2.....	15-2
Instant Messaging Server Cross Domain Searches.....	15-2
Enabling Instant Messaging Server Cross Domain Searches.....	15-3
About Hosted Domains Communication.....	15-3
Enabling Communication Between Hosted Domains.....	15-3
Disabling Communication Between Hosted Domains.....	15-3
Whitelisting and Blacklisting Domains for Hosted Domain Communication.....	15-3
16 Federating Instant Messaging Server with External Servers	
Overview of Federating Multiple Instant Messaging Servers.....	16-1
Securing Server-to-Server Communication.....	16-1
Configuring Federated Communication Between Instant Messaging Servers.....	16-2
Federation Examples.....	16-2
Configuring DNS for XMPP Federation.....	16-3
Configuring DNS for SIP Federation.....	16-3
17 Configuring the XMPP WebSocket Gateway	
About the XMPP WebSocket Gateway.....	17-1
Configuring the XMPP WebSocket Gateway.....	17-1
Managing the XMPP WebSocket Gateway.....	17-3
Configuring XMPP WebSocket Gateway Logging.....	17-3
Configuring XMPP WebSocket Gateway Monitoring.....	17-3
Configuring XMPP WebSocket Gateway Timeout.....	17-4
Configuring XMPP WebSocket Gateway for Server Pool.....	17-4

Enabling Secure Communication Between the XMPP WebSocket Gateway and Instant Messaging Server	17-4
Activating TLS on XMPP WebSocket Gateway	17-4

18 Configuring the XMPP/HTTP Gateway

About the XMPP/HTTP Gateway	18-1
Instant Messaging Server XMPP/HTTP Gateway Configuration Files.....	18-1
Configuring the Instant Messaging Server XMPP/HTTP Gateway.....	18-1
Enabling or Disabling the Instant Messaging Server XMPP/HTTP Gateway.....	18-2
Manually Configuring HTTPBIND	18-2
Configuring Concurrent Requests Handled by the XMPP/HTTP Gateway	18-4
Setting the JEP 124 hold Attribute for Client Requests to the XMPP/HTTP Gateway	18-4
Specifying the Allowed Client Inactivity Time for the XMPP/HTTP Gateway.....	18-4
Setting the content-type HTTP Header for the XMPP/HTTP Gateway	18-5
Setting the Round Trip Delay for the XMPP/HTTP Gateway	18-5
Setting the XMPP/HTTP Gateway Default Response Time	18-5
Configuring an XMPP/HTTP Gateway in an Instant Messaging Server Gateway Pool	18-6
Configuring the List of Key IDs for Supported XMPP/HTTP Gateway Domains	18-6
Configuring the XMPP/HTTP Gateway to Use a Non-default Configuration	18-7
Using Encrypted Passwords.....	18-7
Adding a New Hosted Domain Without Restarting GlassFish Server	18-8
Using StartTLS to Secure Communication Between XMPP/HTTP and IM Server.....	18-8
Managing Logging for the XMPP/HTTP Gateway	18-8
Enabling or Disabling Logging for the XMPP/HTTP Gateway	18-8
Changing the Location of the XMPP/HTTP Gateway Log Configuration File.....	18-9
Setting the XMPP/HTTP Gateway Log File Location on Linux	18-9
Changing the Location of the XMPP/HTTP Gateway Log File.....	18-9
Using a Non-default Log File Location for the XMPP/HTTP Gateway	18-10
Setting the XMPP/HTTP Gateway Logging Level	18-10
XMPP/HTTP Gateway log4j Log Configuration File Syntax.....	18-10
XMPP/HTTP Gateway Log Configuration File Example.....	18-11

19 Configuring External Service Discovery

About External Service Discovery	19-1
Configuring External Discovery.....	19-1

20 Configuring the SIP Gateway

About the SIP Gateway	20-1
SIP Gateway Architecture.....	20-2
Configuring the SIP Gateway.....	20-3
Prerequisites for Configuring the SIP Gateway.....	20-3
Configuring the SIP Gateway in Component Mode.....	20-4
Configuring SIP Gateway in Federation Mode	20-4
Enabling S2S Communication Using TLS and SASL-External.....	20-4
Enabling S2S Communication Using TLS and Dialback	20-5
Enabling S2S Communication Using Plain Text and Dialback	20-5

Configuring Logging for the SIP Gateway	20-5
Configuring the Oracle Communications Converged Application Server	20-5
Testing the SIP Gateway	20-6
Troubleshooting the SIP Gateway	20-6
Configuring DNS for XMPP and SIP Federation	20-6
21 Configuring the SMS Gateway	
About the SMS Gateway	21-1
Configuring the SMS Gateway	21-1
SMS and Server Configuration Properties	21-1
Server-Side Configuration	21-2
Configuring the SMS Gateway by Using the imconfutil Command.....	21-2
Configuring the SMS Gateway by Using the configure Utility	21-3
Client-Side Settings.....	21-4
Starting and Stopping the SMS Gateway.....	21-4
22 Using Calendar Pop-up Reminders	
About Pop-up Reminders	22-1
Pop-up Reminders Operation	22-1
Pop-up Reminders Architectural Flow	22-1
Configuring Calendar Server and Instant Messaging Server to use Pop-ups.....	22-2
Configuring Calendar Pop-ups in a Server Pool.....	22-2
Administering the Calendar Agent	22-2
23 Configuring the Instant Messaging Server Calendar Agent	
Configuring Calendar Agent with Calendar Server.....	23-1
Configuring Instant Messaging Server	23-1
Configuring Instant Messaging Server Calendar Agent with Calendar Server	23-1
Manually Configuring Instant Messaging Server Calendar Agent with Calendar Server	23-2
JMS and Calendar Agent Properties	23-2
24 Displaying Availability Based on Calendar Entries	
Overview of Displaying Instant Messaging Availability Based on Calendar Entries	24-1
Enabling Instant Messaging Availability Based on Calendar Entries	24-1
Configuring Java Message Queue Brokers for Calendar Availability	24-1
25 Using the Web Presence API	
About the Instant Messaging Server Web Presence API.....	25-1
Web Presence API for Requesting Presence Information.....	25-1
web.xml File for the Web Presence API.....	25-1
HTTP GET Requests for Presence Information	25-2
GET Requests for Presence Information on an Individual User	25-2
GET Requests for Presence Information on Multiple Users	25-2
HTTP POST Requests for Presence Information	25-2

JSON Response to Requests for Presence Information.....	25-3
Configuring the Instant Messaging Server Web Presence API	25-3
Configuring an Instant Messaging Server to Recognize the Web Presence API.....	25-5
Configuring and Testing the Web Presence API	25-6
26 Configuring the Instant Messaging Server Web Presence API	
Configuring the Web Presence API.....	26-1
Part III Instant Messaging Server Reference	
27 Configuration File and Directory Structure Overview	
Program Files.....	27-1
Oracle Solaris Location of Program Files.....	27-1
Red Hat Linux and Oracle Linux Location of Program Files	27-1
Server Configuration Files	27-1
Oracle Solaris Location of Server Configuration Files.....	27-1
Red Hat Linux and Oracle Linux Location of Server Configuration Files	27-2
Runtime Directory.....	27-2
Oracle Solaris Location of the Runtime Directory.....	27-2
Red Hat Linux and Oracle Linux Location of the Runtime Directory	27-2
Database Directory.....	27-2
Oracle Solaris Location of the Database Directory.....	27-2
Red Hat Linux and Oracle Linux Location of the Database Directory	27-3
Instant Messaging Server Configuration File.....	27-3
28 Configuration Properties	
iim.conf.xml File Location	28-1
iim.conf.xml File Syntax	28-1
Multiple Server Configuration Properties	28-10
Shoal Configuration Properties.....	28-11
Multiplexor Configuration Properties	28-12
Archive Properties.....	28-13
Watchdog Properties.....	28-15
Monitoring Properties	28-15
Agent Properties	28-16
JMQ Properties	28-17
HTTP/XMPP Gateway Properties	28-17
SMS Integration Properties.....	28-17
XMPP WebSocket Gateway Properties.....	28-18
29 Instant Messaging Server APIs	
Instant Messaging Server APIs Overview.....	29-1
Instant Messaging Server Service API.....	29-1
Service Provider Interfaces.....	29-1
Archive Provider API.....	29-2

Message Conversion API	29-2
Authentication Provider API.....	29-2
Web Presence API	29-2
30 imadmin Command Reference	
imadmin Overview	30-1
imadmin Requirements	30-1
imadmin Location	30-1
imadmin Commands	30-1
imadmin Syntax.....	30-2
imadmin Options	30-3
imadmin Actions	30-3
imadmin Components.....	30-3
31 imconfutil Command Reference	
Syntax	31-1
Options.....	31-1
Example imconfutil Commands.....	31-4
32 iadmin Command Reference	
Overview of the iadmin Command.....	32-1
Syntax	32-1
iadmin Options.....	32-1
iadmin Commands and Command-Specific Options.....	32-1
Example iadmin Commands.....	32-2
33 passwordtool Command Reference	
passwordtool Overview	33-1
Syntax	33-1
Examples	33-2
34 XMPP and HTTP Gateway Configuration Parameters	
httpbind.conf File Location	34-1
httpbind.conf File Syntax	34-1
How Load Balancing Occurs.....	34-2
Instant Messaging Server XMPP/HTTP Gateway Configuration Parameters	34-2
Gateway Domain ID Key Parameters for httpbind.config	34-4

Preface

This guide provides instructions for administering Oracle Communications Instant Messaging Server. Instant Messaging Server enables secure, real-time communication and collaboration, combining presence awareness with instant messaging capabilities such as chat, conferences, and file transfers to create a rich collaborative environment.

Audience

This document is intended for system administrators whose responsibility includes Instant Messaging Server. This guide assumes you are familiar with the following topics:

- Messaging Server and Calendar Server protocols
- Oracle Directory Server Enterprise Edition and LDAP
- System administration and networking
- General deployment architectures

Related Documents

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

- *Instant Messaging Server Installation and Configuration Guide*: Provides instructions for installing and configuring Instant Messaging Server.
- *Instant Messaging Server Release Notes*: Describes the new features, fixes, known issues, troubleshooting tips, and required third-party products and licensing.
- *Instant Messaging Server Security Guide*: Provides guidelines and recommendations for setting up Instant Messaging Server in a secure configuration.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing

impaired.

Part I

Monitoring and Managing Instant Messaging Server

Part I contains the following chapters:

- [Instant Messaging Server Administration Overview](#)
- [Administering Instant Messaging Server Components](#)
- [Using LDAP with Instant Messaging Server](#)
- [Using Oracle Database with Instant Messaging Server](#)
- [Administering Instant Messaging Server End Users](#)
- [Managing Messages for Users](#)
- [Improving Instant Messaging Server Performance](#)
- [Configuring Instant Messaging Server for High Availability](#)
- [Configuring LDAP Failover](#)
- [Managing Archiving in Instant Messaging Server](#)
- [Managing Message Conversion in Instant Messaging](#)
- [Monitoring Instant Messaging Server and Multiplexor](#)
- [Troubleshooting Instant Messaging Server](#)
- [Managing Logging for Instant Messaging Server](#)

Instant Messaging Server Administration Overview

This chapter provides an overview of Oracle Communications Instant Messaging Server administration tasks, and the commands to perform those tasks.

Overview of Instant Messaging Server Administration Tasks

As an Instant Messaging Server administrator, you are responsible for the day-to-day tasks of maintaining and managing Instant Messaging Server and its users. The tasks also include managing Instant Messaging Server components and dependent products.

You perform the following types of tasks as an Instant Messaging Server administrator:

- Manage users and user access to Instant Messaging Server
- Administer Instant Messaging Server components
- Troubleshoot Instant Messaging Server
- Configure and manage messages that interact with Instant Messaging Server
- Configure support for hosted domains
- Manage Instant Messaging Server log files

Instant Messaging Server administration involves managing and maintaining the application and its users, and also the components and servers upon which Instant Messaging Server depends.

About Instant Messaging Server Commands

Following is a list of commands that you use to administer Instant Messaging Server:

- **imadmin**
Use this command to start, stop, and refresh the Instant Messaging server and multiplexor.
- **imconfutil**
Use this command to set, modify, and list Instant Messaging Server configuration properties.
- **iwadmin**

Use this command to create a WAR file or a ZIP file that contains a WAR file and additional files needed for an XMPP Web component.

- **passwordtool**

Use this command to generate encryption keys for passwords, regenerate encryption keys for passwords, generate encrypted passwords, and verify that an unencrypted password matches an encrypted password.

Directory Placeholders Used in This Guide

[Table 1–1](#) lists the placeholders that are used in this guide:

Table 1–1 *Instant Messaging Server Directory Placeholders*

Placeholder	Directory
<i>InstantMessaging_home</i>	Specifies the installation location for the Instant Messaging Server software. The default for both Solaris and Linux is /opt/sun/comms/im .
<i>InstantMessaging_cfg</i>	Specifies the installation location for configuration directory. The default for Solaris is /etc/opt/sun/comms/im/default/config . The default for Linux is /etc/opt/sun/im/default/config .
<i>InstantMessaging_database</i>	Specifies the location for the database directory, if you are using a file-based property store. The default for Solaris is /var/opt/sun/comms/im/default/db . The default for Linux is /var/opt/sun/im/default/db .
<i>InstantMessaging_runtime</i>	Specifies the location for the configurable directory for the files generated by the server at runtime. The default for Solaris is /var/opt/sun/comms/im/default . The default for Linux is /var/opt/sun/im/default .

Administering Instant Messaging Server Components

This chapter explains how to administer Oracle Communications Instant Messaging Server components (server, multiplexor, Calendar agent, watchdog, and SMS Gateway) and perform other administrative tasks, such as changing configuration properties and creating backups.

Overview of Stopping, Starting, Refreshing, and Checking Instant Messaging Server Components

The **imadmin stop** command enables you to:

- Start and stop all Instant Messaging Server components (server, multiplexor, calendar agent, watchdog, and SMS gateway)
- Start and stop an individual Instant Messaging Server component
- Refresh all Instant Messaging Server component configurations
- Refresh an individual Instant Messaging Server component
- Check the status of Instant Messaging Server components

The **imadmin stop** command can be executed only by **root** or a user who has administration rights to the system(s) on which the Instant Messaging server and multiplexor are running. This end user is typically the identity that the server runs as, and is designated during installation:

- On Oracle Solaris - **inetuser**

The **imadmin stop** command is located in the *InstantMessaging_home/sbin* directory.

Starting the Instant Messaging server enables clients to connect to it. Stopping the Instant Messaging server closes all connections and disconnects all clients.

Using Service Management Framework (SMF)

Instant Messaging Server supports the Service Management Framework (SMF) for stopping and starting Instant Messaging Server. On supported platforms, an SMF service is registered when you install Instant Messaging Server. You can use either the **svcadm** command or the Instant Messaging Server **imadmin stop** command to start and stop Instant Messaging Server processes.

To start and stop Instant Messaging Server by using the **svcadm** command:

```
svcadm enable svc:/application/sunim
```

```
svcadm disable svc:/application/sunim
```

To check the status of the Instant Messaging Server service:

```
svcs sunim
```

SMF related log messages can be found in the `/var/svc/log/application-sunim:etc-opt-SUNWiim-default.log` file.

Starting Instant Messaging Server Components

You can start all the components together or a single component separately.

Use the **imadmin stop** command with the **start** option to start the Instant Messaging Server, multiplexor, Calendar agent, watchdog, and SMS Gateway, depending on which components are enabled.

Starting All Components

To start all Instant Messaging Server components, enter the following command:

```
imadmin start
```

If both server and multiplexor are enabled, this command first starts the Instant Messaging server, and then starts the multiplexor.

If the watchdog is enabled (default), this command starts the watchdog, then the watchdog reads the configuration file and starts the Instant Messaging server and/or multiplexor as necessary.

Starting a Single Component

To start a single Instant Messaging Server component, enter the **imadmin start** command with an argument that designates the component as follows:

- Server:

```
imadmin start server
```

- Multiplexor:

```
imadmin start multiplexor
```

- Calendar agent:

```
imadmin start agent-calendar
```

- Watchdog:

```
imadmin start watchdog
```

- SMS Gateway:

```
imadmin start sms-gateway
```

Stopping Instant Messaging Server Components

You can stop all the components together or a single component separately.

Use the **imadmin stop** command with the **stop** option to stop the Instant Messaging server, multiplexor, Calendar agent, watchdog, and SMS Gateway, depending on which components are enabled.

Stopping All Components

To stop all Instant Messaging Server components, enter the following command:

```
imadmin stop
```

If the watchdog is running, **imadmin stop** brings the watchdog down first, and then stops the server and the multiplexor, or just the multiplexor if that is running separately.

This command stops the server, multiplexor, Calendar agent, watchdog, and SMS Gateway, terminates all end user connections, and disconnects any inbound and outbound servers configured.

Stopping a Single Component

To stop a single Instant Messaging Server component, enter the **imadmin stop** command with an argument that designates the component as follows:

- Server:

```
imadmin stop server
```

- Multiplexor:

```
imadmin stop multiplexor
```

- Calendar agent:

```
imadmin stop agent-calendar
```

- Watchdog:

```
imadmin stop watchdog
```

Registering and Unregistering Service Management Framework

Service Management Facility (SMF) is a feature of the Solaris Operating System that creates a supported, unified model for services and service management on each Solaris system. It is a mechanism to define, deliver, and manage long-running application services for Solaris.

Registering SMF

There is no need to explicitly register SMF. The Instant Messaging Server initial configuration process registers SMF. However, if need be, at the command line, enter the following command:

```
imadmin smf-register
```

You must first stop services before running the command.

Unregistering SMF

To unregister SMF, enter the following command:

```
imadmin smf-unregister
```

Refreshing Component Configuration

Use the **imadmin stop** command with the **refresh** option to stop and restart an individual Instant Messaging Server component and refresh that component's configuration. You can refresh all the components together or a single component

separately. Whenever you change a configuration property in the **iim.conf.xml** file, you also need to refresh the configuration.

Refreshing All Components

To refresh all Instant Messaging Server components, enter the following command:

```
imadmin refresh
```

This command stops the server, multiplexor, Calendar agent, watchdog, and SMS Gateway, terminates all end user connections, and disconnects any inbound and outbound servers configured.

If the watchdog is running, **imadmin stop** brings the watchdog down first, and then stops the server and/or the multiplexor. The command then starts the watchdog, which reads the configuration file and starts the Instant Messaging server and/or multiplexor as necessary.

Refreshing a Single Component

To refresh a single Instant Messaging Server component, enter the **imadmin refresh** command with an argument that designates the component as follows:

- Server:

```
imadmin refresh server
```

- Multiplexor:

```
imadmin refresh multiplexor
```

- Calendar agent:

```
imadmin refresh agent-calendar
```

- Watchdog:

```
imadmin refresh watchdog
```

Checking the Status of Instant Messaging Server Components

You can check the status of all the components together or a single component separately by using the **imadmin stop** command with the **status** option. This command returns results in the following format: *Component [status]*.

For example:

```
Server [UP]
```

```
Multiplexor [UP]
```

```
Agent:calendar [DOWN]
```

```
Watchdog [UP]
```

To check the status of all components, enter the following command:

```
imadmin status
```

To check the status of a single component, enter the **imadmin status** command with an argument that designates the component as follows:

- Server:

```
imadmin status server
```

- Multiplexor:

```
imadmin status multiplexor
```

- Calendar agent:

```
imadmin status agent-calendar
```

- Watchdog:

```
imadmin status watchdog
```

Changing Instant Messaging Server and Multiplexor Configuration Properties

Instant Messaging Server stores configuration properties in the **iim.conf.xml** file. See ["Configuration Properties"](#) for a complete list of configuration properties.

To change configuration properties, use the **imconfutil** command then refresh the Instant Messaging Server configuration. If you change a multiplexor property, you only need to refresh the multiplexor as follows:

```
imadmin refresh multiplexor
```

Changing Configuration Properties

To change the configuration properties:

1. Run the **imconfutil** command to set the configuration properties.
2. Refresh the configuration:

```
imadmin refresh
```

Backing Up Instant Messaging Server Data

Instant Messaging Server does not provide its own disaster recovery tools. Use your site's backup system to back up the configuration and database directories periodically.

Backup Information

The types of Instant Messaging Server information that needs to be backed up are:

- Configuration information
- Instant Messaging Server end user data

The configuration information is stored in the configuration directory (*InstantMessaging_cfg*). The Instant Messaging Server data is stored in the database directory (*InstantMessaging_database*). Default paths are described in ["Configuration File and Directory Structure Overview"](#).

Performing a Backup

While the configuration information does not change frequently, the Instant Messaging Server end-user data changes rapidly and to prevent any loss of end-user data you should back up the Instant Messaging Server end-user data on a periodic basis. You

must perform the backup before running the installation program and the uninstallation program.

To back up the end user data and the configuration information you do not have to stop the Instant Messaging server as all the disk commits by the server are automatically performed.

Restoring Backup Information

The end-user data and the configuration information need to be restored when there is a disk failure and all the end-user data and the configuration information is lost.

To restore the end-user data from backup:

1. Change to the *InstantMessaging_runtime* directory.

See "[Configuration File and Directory Structure Overview](#)" for information on locating *InstantMessaging_runtime*.

2. Stop the Instant Messaging server:

```
imadmin stop
```

3. Copy the backed-up data to the *InstantMessaging_database* directory.

Be sure to maintain the directory structure of the backed-up data.

4. Verify the permissions and owner of the newly restored data.

The files should be owned by the Instant Messaging Server system user. See the topic on creating a UNIX system user and group in *Instant Messaging Server Installation and Configuration Guide*. Permissions should be set as follows:

- a. Files: **600** (indicating read and write permissions for owner only)
- b. Directories: **700** (indicating read, write, and execute permissions for owner only)

Refer to your operating system documentation for information on changing permissions and owners.

5. Start the Instant Messaging server:

```
imadmin start
```

Using LDAP with Instant Messaging Server

This chapter covers aspects of using LDAP with Oracle Communications Instant Messaging Server.

Managing LDAP Access Configuration for Instant Messaging Server

This section describes how Instant Messaging Server uses LDAP.

Overview of How Instant Messaging Server Uses LDAP

All deployments of Instant Messaging Server require an Oracle Directory Server Enterprise Edition (Directory Server). Instant Messaging Server uses the Directory Server to perform end-user authentication and to search for end users.

The default Instant Messaging Server configuration makes the following assumptions regarding the LDAP schema used by Directory Server:

- End user entries are identified by the **inetOrgPerson** object class.
- Group entries are identified by the **groupOfUniqueNames** or **groupofURLs** object class.
- 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).

You can change these default settings by running the **imconfutil** command to modify the appropriate configuration properties.

Caution: Some user attributes might contain confidential information. Ensure that your directory access control is set up to prevent unauthorized access by non-privileged users. Refer to your directory documentation for more information.

Searching the Directory Anonymously

Instant Messaging Server must be able to search the directory to function correctly. If your directory is configured to be searchable by anonymous users, Instant Messaging Server has the capability to search the directory. If the directory is not readable or searchable by anonymous users, you must take additional steps to configure the **iim.conf.xml** file with the credentials of a user ID that has at least read access to the directory. These credentials consist of:

- A distinguished name (dn)
- The password of the distinguished name (dn)

Enabling the Server to Conduct Directory Searches as a Specific End User

To enable Instant Messaging Server to conduct directory searches as a specific end user:

1. Identify values for the following Instant Messaging Server configuration properties.
 - **iim_ldap.usergroupbinddn** - Specifies the distinguished name (dn) to use to bind to the directory for searches.
 - **iim_ldap.usergroupbindcred** - Specifies the password to use with the distinguished name (dn).

For example:

```
iim_ldap.usergroupbinddn="cn=iim server,o=i-zed.com"
iim_ldap.usergroupbindcred=secret
```

Note: You do not have to use administrator-level credentials with write-level access. All that is necessary is read access to the domain tree. Thus, if there is an LDAP user with read-level access, use its credentials instead. This is a safer alternative as it does not force you to disseminate the administrator-level credentials.

2. Run the **imconfutil** command to modify the configuration properties.

If the **iim_ldap.usergroupbinddn** and **iim_ldap.usergroupbindcred** properties do not appear in the **iim.conf.xml** file, add them.

For example:

```
imconfutil set-prop iim_ldap.usergroupbinddn="cn=Directory Manager" iim_
ldap.usergroupbindcred=password -c InstantMessaging_home/config/iim.conf.xml
```

Using the mail Attribute for User Authentication

By default, Instant Messaging Server uses the **uid** attribute for user authentication. You can change the default so that users can authenticate based on their email address.

To configure Instant Messaging Server to enable user authentication by using the **mail** attribute:

1. Use the **imconfutil** command to set the **mail** attribute to use for user IDs as the value for the **iim_ldap.useruidattr** property:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml iim_
ldap.useruidattr=mail
```

2. Add the index directive to the indexing rules in LDAP:

```
index mail eq
```

Using LDAP Groups

You can configure Instant Messaging Server so that end users can send a message to an LDAP group, which can be either dynamic or static. Also, you can assign/affiliate LDAP groups as members of a restricted chat room.

LDAP distinguishes between dynamic and static groups as follows:

- LDAP dynamic group: Membership, rather than being maintained explicitly in a list, is determined by search criteria using an LDAP URL. Dynamic groups use the **groupOfURLs** object class and the **memberURL** attribute to define LDAP URLs with the criteria (search base, scope, and filter) to be used for determining members of the group.
- LDAP static group: A static group is one whose entry contains a membership list of explicit DNs. You can define a static group by using the **groupOfUniqueNames** object class and by explicitly specifying the member DNs using the **uniqueMember** attribute.

In Directory Server and some other LDAP servers, dynamic groups filter end users based on their DN and include them in a single group. The dynamic groups are defined in Directory Server by the **groupOfURLs** object class.

To enable end users to view the dynamic groups in search results and add them to their contact list, you must include **groupOfURLs** objects in search results.

You can assign or affiliate LDAP groups as members of a restricted chat room. When Instant Messaging Server creates the multiuser chat room, it loads the chat room's affiliations from LDAP. Instant Messaging Server determines if the user is a member of any of the groups authorized for the multiuser chat room. Instant Messaging Server then allows the users to join if they are members and otherwise forbids users from joining. You use the **iim_server.enablegroupsinconference** property to enable and disable multiuser chat. When set to **true**, this property enables groups for multiuser chat. When set to **false**, this property disables groups for multiuser chat. By default, groups for multiuser chat is disabled.

Configuring Instant Messaging Server to Use LDAP Groups

To configure Instant Messaging Server to use LDAP groups:

1. If you have not already done so, create the LDAP group to be used for group messaging. See "Managing Groups" in *Sun Java System Directory Server Enterprise Edition 6.0 Administration Guide* at:

<http://docs.oracle.com/cd/E19693-01/819-0995/bcajq/index.html>

2. Set the **iim_server.group.servicename** property, if the service name for group messaging is to be changed from the default name groups.

For example:

```
imconfutil set-prop iim_server.group.servicename=mygroups -c InstantMessaging_
home/config/iim.conf.xml
```

3. Set the appropriate configuration property, depending on if you want to search for dynamic or static groups.

- To search for dynamic groups, set the following properties:

```
imconfutil set-prop iim_
ldap.groupbrowsefilter="( | (& ((objectclass=groupofurls) (cn={0})))" -c
InstantMessaging_home/config/iim.conf.xml
imconfutil set-prop iim_ldap.groupclass=groupofurls -c InstantMessaging_
```

```
home/config/iim.conf.xml
```

- To search for static groups, set the following properties:

```
imconfutil set-prop iim_
ldap.groupbrowsefilter="( | (&(objectclass=groupofuniquenames) (cn={0})) )" -c
InstantMessaging_home/config/iim.conf.xml
imconfutil set-prop iim_ldap.groupclass=groupofuniquenames -c
InstantMessaging_home/config/iim.conf.xml
```

Note: Static groups can also be inherited from **groupofnames** object class, and their members listed using member attribute. However, the search filters for static groups must be modified accordingly. By default, the member attribute is not used as the membership attribute of a static group. Hence, the property must be set to **iim_ldap.groupmemberattr=member** to use member attribute.

Do not include line breaks within a single line. The attribute and object class names are configurable. By default, the **memberOfUrls** attribute is used as the membership attribute of a dynamic group. If you want to use an attribute name other than **memberOfUrls**, set the **iim_ldap.groupmemberurlattr** option to the attribute name you want to use.

- To search for both dynamic and static groups, set the following properties:

```
imconfutil set-prop iim_
ldap.groupbrowsefilter="( | (&(objectclass=groupofuniquenames) (cn={0})) (&(obj
ectclass=groupofurls) (cn={0})) )" -c InstantMessaging_
home/config/iim.conf.xml
imconfutil set-prop iim_ldap.groupclass=groupofurls,groupofuniquenames -c
InstantMessaging_home/config/iim.conf.xml
```

4. To search for static groups having **groupofnames** object class, set the following properties:

```
imconfutil set-prop iim_
ldap.groupbrowsefilter="(&(objectclass=groupofnames) (cn={0}))" -c
InstantMessaging_home/config/iim.conf.xml
imconfutil set-prop iim_ldap.groupclass=groupofnames -c InstantMessaging_
home/config/iim.conf.xml
imconfutil set-prop iim_ldap.groupmemberattr=member -c InstantMessaging_
home/config/iim.conf.xml
```

5. To send a message to a group, see ["Using Group Messaging"](#).

Using Group Messaging

To use group messaging:

1. In the client's chat window, type the group's full Jabber ID in the form *groupName@group.domainname* in the To tab. For example:
testGroup@mygroups.example.com
2. Type the message and click send.

Configuring Chat Room Membership Based on LDAP Group Membership

To configure chat room membership based on LDAP group membership:

1. See "[Configuring Instant Messaging Server to Use LDAP Groups](#)" for instructions on how to create an LDAP group to use for the chat room.
2. To enable LDAP groups for multiuser chat, set the **iim_server.enablegroupsinconference** property to **true**.

```
imconfutil set-prop iim_server.enablegroupsinconference=true -c
InstantMessaging_home/config/iim.conf.xml
```

Using Chat Rooms Based on LDAP Group Membership

To use LDAP group-based chat rooms:

1. Create a persistent member-only chat room.

Member-only chat rooms enable only those users who are part of the chat room's member list to join the chat room. Create a new chat room and configure the chat room to be a persistent member-only chat room by selecting **Persistent** and **Restricted** options in the chat room configuration window.

2. Assign a group affiliation to the chat room.

To enable members of a particular group to join and participate in a member-only room, the group must be affiliated to the room. To do so, the chat room administrator adds the group to the chat room's member list. By default, the creator of the chat room is its administrator.

- a. To add the group to the chat room's member list, run the following command in the chat window:

```
/affiliate member testGroup@mygroups.example.com
```

- b. To verify that the group was added successfully to the member list, run the following command:

```
/affiliate member
```

Users of the group should now be able to join and participate in the chat room.

Converting Associated Domain to Canonical Domain

Directory Server can contain a domain entry with an **associatedDomain** attribute. If a user tries to login by using the **associatedDomain**, instead of using the canonical domain, Instant Messaging Server can convert the domain part as needed, from associated domain to canonical domain, before processing the request. Instant Messaging Server supports the following operations for this conversion:

- Authentication
- Presence subscription
- Multiuser chat invite

To enable conversion of the associated domain to the canonical domain:

1. Configure the **iim_server.inboundpacketfilters** property:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.inboundpacketfilters=oracle.communications.ucs.ocim.filters.impl.Cannoni
calDomainConversionFilter
```

2. Set the filter to be used for the **associatedDomain** attribute during an LDAP search:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
ldap.schema2.domain_
filter=" (&(objectClass=sunManagedOrganization) (| (sunPreferredDomain={0}) (associ
atedDomain={0}))) "
```

Using Oracle Database with Instant Messaging Server

This chapter describes how to install and configure Oracle Database to store multiuser chat history for Oracle Communications Instant Messaging Server. Currently, you can store only multiuser chat history and no other data in Oracle Database.

Overview of Using Oracle Database for Storing Messages

By default Instant Messaging Server uses Directory Server to store user properties and data, which includes multiuser chat history. Instead of using Directory Server, you can use Oracle Database to store multiuser chat history.

Installing and Creating an Oracle Database Instance for Instant Messaging Server

Installing and creating an Oracle Database instance for Instant Messaging Server involves the following high-level steps:

1. Downloading and installing Oracle Database
2. Creating the database instance
3. Creating the database user
4. Creating the Instant Messaging Server tables
5. Downloading and installing Oracle JDBC drivers
6. Configuring Instant Messaging Server properties

You install and create the Oracle Database instance by using the Oracle Universal Installer.

To install and create an Oracle Database instance for Instant Messaging Server:

1. Download Oracle Database from the Oracle Technology Network website at:
<http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index.html>
2. To install and create a new Oracle Database instance, follow the instructions in the installation guide for Oracle Database for your operating system, at:
 - Oracle Database 11g, Release 2:
http://www.oracle.com/pls/db112/portal.portal_db?selected=11
 - Oracle Database 12:

http://docs.oracle.com/database/121/nav/portal_11.htm

3. Create the Instant Messaging Server database user.

In the following example, the database user is **imuser**. You use this user, and its associated password, when you set the **iim_server.jdbc.connection.username** and **iim_server.jdbc.connection.password** properties later in this procedure.

```
$ sqlplus SYSTEM as sysdba

alter session set "_ORACLE_SCRIPT"=true;
create user demo identified by "imuser";
GRANT RESOURCE, CONNECT TO imdatabase;
GRANT UNLIMITED TABLESPACE TO imdatabase;
```

4. Run the following command to create the Instant Messaging Server tables:

```
sqlplus uid/passwd < InstantMessaging_home/lib/ocim_oracle.sql
```

5. Download and install Oracle JDBC drivers from the Oracle Database Drivers Download web site at:

<http://www.oracle.com/technetwork/database/features/jdbc/jdbc-drivers-12c-download-1958347.html>

6. Use the **imconfutil** command to set the following Instant Messaging Server configuration properties.

```
iim_server.classpath=/opt/instantclient_12_1/ojdbc7.jar
iim_server.jdbc.connection.url=jdbc:oracle:thin:@host_name:1521:orcl
iim_server.jdbc.connection.username=user
iim_server.jdbc.connection.password='password'
iim_server.jdbc.connection.initialCapacity=10
iim_server.jdbc.connection.maxCapacity=100
iim_
server.storage.providers="oracle.communications.ucs.ocim.provider.db.DBConferen
ceHistoryStore"
iim_server.conference.history.persist=true
```

where:

- *host_name* is the database host
- *user* is the database user for Instant Messaging Server that you previously created
- *password* is the password for the database user

Administering Instant Messaging Server End Users

This chapter describes how to provision Oracle Communications Instant Messaging Server users.

Overview of Administering Instant Messaging Server End Users

Instant Messaging Server does not provide bulk user provisioning tools. You must use a directory bulk provisioning tool for provisioning multiple Instant Messaging Server end users. By default, Instant Messaging Server does not provide specific commands to add, modify, or delete Instant Messaging Server end users. However, you can customize Instant Messaging Server to enable users to add themselves to the directory.

Because Instant Messaging Server users reside in LDAP, you cannot prevent them from using Instant Messaging Server. The only way to prevent end users from using Instant Messaging Server is to delete them from the directory or inactivate their user accounts in the directory. Keep in mind that doing this also prevents the user from binding to the directory.

The administrator can manage Instant Messaging Server end users by using the Instant Messaging Server Administrator Access Control mechanism. For more information, see the topic on controlling privileges in *Instant Messaging Server Security Guide*.

Registering New Instant Messaging Server Users

You can customize Instant Messaging Server to allow new user registration. When a user registers, Instant Messaging Server uses the information provided during registration to perform an **ldapadd** operation to create a user entry in the directory.

Configuring Instant Messaging Server to Enable New User Registration

To configure Instant Messaging Server to allow new user registration you must add the configuration properties listed in [Table 5-1](#).

Table 5-1 Instant Messaging Server New User Registration Configuration Properties

Property	Description
iim.register.enable	If true , the server allows new Instant Messaging Server end users to register themselves (add themselves to the directory).

Table 5–1 (Cont.) Instant Messaging Server New User Registration Configuration

Property	Description
iim_ldap.register.basedn	If self-registration is enabled, the value of this property is the DN of the location in the LDAP directory in which person entries are stored. For example: ou=people,dc=siroe,dc=com .
iim_ldap.register.domain	The domain to which new users will be added. For example, directory.siroe.com .

Allowing New User Registration Using Policy Module iim_ldap

To configure Instant Messaging Server to allow new user registration using **iim_ldap** (**iim.policy.modules = iim_ldap**):

1. Use the **imconfutil** command to add the configuration properties and appropriate values as described in "Configuration Properties".

For example:

```
imconfutil set-prop -c InstantMessaging_home/config/iim.conf.xml
iim.register.enable=true iim_
ldap.register.basedn="ou=people,o=india.sun.com,dc=india,dc=sun,dc=com" iim_
ldap.register.domain=india.sun.com
```

2. Refresh the server configuration by using the **imadmin** command.

```
imadmin refresh server
```

Allowing New User Registration Using Policy Module schema1 or schema2

To configure Instant Messaging Server to allow new user registration using policy module schema 1 or schema 2 (**iim.policy.modules = iim_ldap_schema1** or **iim.policy.modules = iim_ldap_schema2**):

1. Use the **imconfutil** command to add the configuration properties and appropriate values as described in "Configuration Properties".

For example:

```
imconfutil set-prop -c InstantMessaging_home/config/iim.conf.xml
iim.register.enable=true
```

2. Refresh the server configuration by using the **imadmin** command.

```
imadmin refresh server
```

Storing Instant Messaging Server User Properties in LDAP

By default Instant Messaging Server stores user properties in LDAP. You must run the **imadmin assign_services** command to add required object classes to user entries in the directory. These object classes are used by Instant Messaging Server to store user properties in user entries.

Caution: Some user attributes may contain confidential information. Ensure that your directory access control is set up to prevent unauthorized access by non-privileged users. Refer to your directory documentation for more information.

To store Instant Messaging Server user properties in LDAP:

1. Ensure that the **iim.policy.modules** configuration property has a value of **iim_ldap**, and that the **iim.userprops.store** configuration property has a value of **ldap**.

For example:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml get-prop|grep  
iim.policy.modules
```

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml get-prop|grep  
iim.userprops.store
```

If the values are not set, use the **imconfutil** command to set them.

2. Run **imadmin assign_services**:

```
imadmin assign_services
```

The **imadmin** command checks the value of the **iim.policy.modules** property.

3. Enter the Bind DN and password you want **imadmin** to use to bind to the directory.

The Bind DN should have sufficient credentials to modify the directory schema, for example the Directory Manager DN.

4. Enter the Base DN under which user entries are stored.

Next, **imadmin** adds **sunIMUser**, and **sunPresenceUser** object classes to the user entries in the organization you specified.

Managing Messages for Users

This chapter describes how you can manage messages that are sent to offline Oracle Communications Instant Messaging Server users so they receive queued messages the next time they log in. The messages can be sent as an SMS or forwarded as an email. This chapter also describes how to enable push notifications of offline messages, and how to enable message carbons to keep all online devices for a user synchronized.

About Managing Instant Messaging Server Messages for Offline Users

When you enable offline chat message delivery on Instant Messaging Server, regular instant messages (chat messages) that are sent to offline users are not discarded. They are available on Instant Messaging Server and delivered to the user when the user comes online. You enable offline chat message delivery at the deployment level or at the domain level by using the whitelisting facility. You can disable offline chat message delivery at the domain level by using the blacklisting facility.

Alerts are managed separately. Similar to offline chat messages, Instant Messaging Server stores the alerts for the offline recipient, and delivers them when the user logs in next time, as long as the user has been configured to receive offline alerts during the next login. You enable or disable offline alerts at the user level.

Note: If you install Oracle Communications Calendar Server with Instant Messaging Server, you can configure your deployment such that you receive Instant Messaging Server alerts about your calendar tasks and events as pop-up messages. As long as you are logged in to Instant Messaging Server (online), you receive Calendar Server HTML pop-up reminders on your desktop. If you are offline, you receive the alerts the next time you login, as long as have configured Instant Messaging Server to receive offline alerts during the next login.

For more information on calendar alerts, see the topic on planning your installation in *Instant Messaging Server Installation and Configuration Guide*.

See "[Configuring the SMS Gateway](#)" for information about SMS forwarding.

About Delivering Offline Messages

You can enable support for offline messages in Instant Messaging Server. Instant Messaging Server stores the messages intended for an offline recipient. The server then delivers the offline messages once the recipient comes online. The implementation of the offline message support is based on the XEP-160 guidelines, and supports the entire deployment to cover domains.

Instant Messaging Server supports offline message delivery of one-to-one chat. The offline message support for a multiuser chat is available in persistent chat rooms. You can set the maximum number of chat messages you want to receive. You can use the same properties to whitelist or blacklist specific domain names for which this capability can be enabled. The messages are queued for offline delivery in the Instant Messaging Server data directory on the file system of the host. For more information, see the topic on support topics in *Instant Messaging Server Installation and Configuration Guide*. Also, see *Best Practices for Handling Offline Messages* at:

<http://xmpp.org/extensions/xep-0160.html>

Table 6–1 shows the configuration properties used to enable support for offline messages:

Table 6–1 Configuration Properties for Enabling Support of Offline Messages

Property	Default Value	Description
iim_server.deliverofflinechat	false	Determines whether the capability is on or off. To enable the feature for the entire deployment, set the iim_server.deliverofflinechat property to true , and do not set the deliverofflinechat.domain property. To disable the feature for the entire deployment, set the iim_server.deliverofflinechat property to false , and do not set the deliverofflinechat.domain property.
deliverofflinechat.domain	None	Used to blacklist or whitelist a domain. To blacklist a domain, set the iim_server.deliverofflinechat property to true , and set the deliverofflinechat.domain property to the list of domains to be blacklisted. To whitelist a domain, set the iim_server.deliverofflinechat property to false , and set the deliverofflinechat.domain property to the list of domains to be whitelisted.
deliverofflinechat.maxsize	50	Determines the maximum queue size related to the Receiver, and must be a positive integer.

Note: All peers or machines in a server pool environment store the messages locally on the users' file system. If a machine is unavailable, the stored messages are not available for delivery to the user. The messages are delivered only when the machine is up, and the recipient user is online.

See "Configuration Properties" for more information on the complete list of Instant Messaging Server configuration properties.

Enabling Offline Messaging

To enable offline messaging:

1. Set the configuration properties by using the **imconfutil** command for the following scenarios:
 - a. To enable the feature for all, set **iim_server.deliverofflinechat** to **true**, and do not set **deliverofflinechat.domain**. For example:


```
imconfutil set-prop iim_server.deliverofflinechat=true -c InstantMessaging_
home/config/iim.conf.xml
```
 - b. To disable the feature for all, set **iim_server.deliverofflinechat** to **false**, and do not set **deliverofflinechat.domain**. For example:

```
imconfutil set-prop iim_server.deliverofflinechat=false -c
InstantMessaging_home/config/iim.conf.xml
```

- c. To whitelist a domain, set **iim_server.deliverofflinechat** to **false**, and set **deliverofflinechat.domain** to *comma_separated_list_of_domains_to_be_whitelisted*. For example:

```
imconfutil set-prop iim_server.deliverofflinechat=false -c
InstantMessaging_home/config/iim.conf.xml
```

```
imconfutil set-prop deliverofflinechat.domain="comma_separated_list_of_
domains_to_be_whitelisted" -c InstantMessaging_home/config/iim.conf.xml
```

- d. To blacklist a domain, set **iim_server.deliverofflinechat** to **true**, and set **deliverofflinechat.domain** to *comma_separated_list_of_domains_to_be_denied*. For example:

```
imconfutil set-prop iim_server.deliverofflinechat=true -c InstantMessaging_
home/config/iim.conf.xml
```

```
imconfutil set-prop deliverofflinechat.domain="comma_separated_list_of_
domains_to_be_denied" -c InstantMessaging_home/config/iim.conf.xml
```

2. Restart Instant Messaging Server.

About Push Notifications for Offline Messages

You can enable support for push notification of offline messages by using Instant Messaging Server's publish-subscribe (pub-sub) service. The pub-service is a notification service that uses XMPP. Instant Messaging Server uses a pub-sub *node*, also referred to as a *topic*, to store information about offline messages. Instant Messaging Server publishes the offline notification messages as an *event* to the pub-sub node. A subscriber in the form of an XMPP component can register with the node to receive notification of offline messages. The subscriber is a component outside the scope of Instant Messaging Server. The requirement for the subscriber is that it communicate by using XMPP and that you configure it to communicate with Instant Messaging Server.

For scalability reasons, deploy the subscriber as its own component, and configure it to subscribe to the pub-sub node created in Instant Messaging Server. The Instant Messaging Server host is both the owner and publisher of the pub-sub node. Access to pub-sub nodes is open for component connections but closed for client connections.

Table 6–2 shows the configuration properties used to enable support for offline push notification messages:

Table 6–2 Configuration Properties for Enabling Support of Push Notifications for Offline Messages

Property	Default Value	Description
iim_server.pubsub.customnode.enable	false	Determines whether the capability is on or off. To enable the feature for the entire deployment, set the iim_server.pubsub.customnode.enable property to true . To disable the feature for the entire deployment, set the iim_server.pubsub.customnode.enable property to false .
iim_server.pubsub.customnode.offline message	None	Used to set the topic name for the pub-sub node.

For more information on the complete list of Instant Messaging Server configuration properties, see ["Configuration Properties"](#).

Enabling Push Notifications for Offline Messages

This task assumes the use of a third-party application as the component that subscribes to the pub-sub node. Such a component is outside the scope of Instant Messaging Server.

To enable push notifications for offline messages:

1. Enable support for offline messaging.
See ["Enabling Offline Messaging"](#) for more information.
2. Use the **imconfutil** command to set the following properties for the publish-subscribe service and the pub-sub node:

```
iim_server.pubsub.customnode.enable=true  
iim_server.pubsub.customnode.offlinemessage=offline_node_name
```

where *offline_node_name* is the name of the pub-sub node, or topic.

3. Configure the component that is subscribing to *offline_node_name*.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml add-component  
id=PubSubConsumer jid="pubsubcomponent" password=password
```

4. Restart Instant Messaging Server.

About Message Carbons

Using message carbons, you can configure Instant Messaging Server so that the carbon-enabled devices simultaneously display up-to-date conversations with the most current messages on both sides of the communication. Thus, if users switch between devices, such as desktop computers, smartphones, and tablets, all their devices display the complete communication that has taken place.

While message carbons can be used to keep all online devices synchronized, you must use archive management to synchronize conversation history for offline devices. See ["About Archived Messages Retrieval"](#) for more information.

Enabling Message Carbons

To enable message carbons:

1. Set the **iim_server.carbon.enable** property to **true**:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_  
server.carbon.enable="true"
```

2. Restart Instant Messaging Server.

Improving Instant Messaging Server Performance

This chapter describes how to enhance the tuning and performance of Oracle Communications Instant Messaging Server.

Tuning Instant Messaging Server Memory

Use the most current version of Java suitable for your operating system and version of Instant Messaging Server. For information about Java versions, see the topic on installing Java in *Unified Communication Suite Installation and Configuration Guide*.

Instant Messaging Server uses the `iim.jvm.maxmemorysize` configuration property to set the maximum size of the Java Virtual Machine (JVM) heap to allocate. The default value of this property is 256 Mbytes. However, a large active deployment of Instant Messaging Server needs more memory. Determining the amount of memory to allocate for the Instant Messaging server depends on the number of concurrent active users that you must support.

Additional load per user, use of additional Instant Messaging Server services like news or file transfer, and use of features such as message filters, archiving, or TLS require more memory. You should perform load profiling of typical user activity before deploying Instant Messaging Server into a production environment. Contact Oracle Support Services for more information about load profiling an Instant Messaging Server deployment.

Tuning Java Virtual Machine Options

[Table 7–1](#) describes the Java Virtual Machine (JVM) options that you should set on both the Instant Messaging Server and multiplexor.

Table 7–1 JVM Options for Instant Messaging Server and Multiplexor

JVM Option	Description
<code>-XX:+DisableExplicitGC</code>	By default calls to <code>System.gc()</code> are enabled (<code>-XX:-DisableExplicitGC</code>). Use <code>-XX:+DisableExplicitGC</code> to disable calls to <code>System.gc()</code> . The JVM still performs garbage collection when necessary.
<code>-XX:+UseG1GC</code>	Use the Garbage First (G1) Collector.
<code>-XX:MaxGCPauseMillis=n</code>	Sets a target for the maximum garbage collection pause time. This is a soft goal, and the JVM makes its best effort to achieve it.

Table 7–1 (Cont.) JVM Options for Instant Messaging Server and Multiplexor

JVM Option	Description
<code>-XX:GCPauseIntervalMillis</code>	Sets the time interval over which garbage collection pauses totaling up to <code>MaxGCPauseMillis</code> can take place.
<code>-XX:+PrintGCTimeStamps</code>	Prints timestamps at garbage collection.
<code>-XX:+PrintGCDetails</code>	Prints more details at garbage collection.
<code>-XX:+PrintGCDateStamps</code>	Prints date stamps at garbage collection events.
<code>-Xloggc:filename</code>	Logs garbage collection verbose output to specified file. The verbose output is controlled by the normal verbose garbage collection flags.

Instant Messaging Server and Multiplexor Thread Pooling and Service Port Configuration

[Table 7–2](#) lists the set of properties that you configure to tailor the size and behavior of thread pools used to service client-to-server and server-to-server requests.

Table 7–2 Thread Pooling and Service Port Properties

Property	Description	Default Value
<code>iim_server.maxthreads</code>	Maximum number of threads for the default thread pool.	100
<code>iim_server.threadpool.capacity</code>	Capacity of the default thread pool.	-1 (no limit)
<code>iim_server.maxqueues.process</code>	Maximum number of threads for process queue.	40
<code>iim_ldap.groupchatstorage.maxpoolsize</code>	Maximum number of threads for group chat history. This property is required if group chat persistent is enabled. (<code>iim_server.conference.history.persist</code>).	40
<code>iim_ldap.groupchatstorage.queuesize</code>	Capacity of group chat thread pool queue. This property is required if group chat persistent is enabled.	120000

You can configure thread pool size and behavior used to service client-to-server and server-to-server requests. If these thread pools are not configured, the server uses the default thread pool. These thread pools, combined with the associated service ports, can improve the throughput of an Instant Messaging server.

[Table 7–3](#) shows the defined thread pools and service port configurations.

Table 7–3 Defined Thread Pools and Service Port Configurations

Service Port	Thread Pool ID	Use	Thread Pool Size	Buffer Size (Bytes)
c2s	worker-in	All multiplexor-to-server inbound communications.	5	1024000

Table 7–3 (Cont.) Defined Thread Pools and Service Port Configurations

Service Port	Thread Pool ID	Use	Thread Pool Size	Buffer Size (Bytes)
c2s	worker-out	All multiplexor-to-server outbound communications	5	1024000
s2s	worker-in	All server-to-server inbound communications. If the port allows server-to-server inbound communications, Instant Messaging Server uses this thread pool.	12	1024000
s2s	worker-out	All server-to-server outbound communications. If the port allows server-to-server outbound communications, Instant Messaging Server uses this thread pool.	12	1024000

Creating Service Port Configurations

To create service port configurations:

1. Run the following commands to create defined thread pools.

The first two commands configure the thread pool for mux-to-server inbound and outbound communications, and the last two commands configure the thread pool for server-to-server inbound and outbound communications.

```
imconfutil -u add-server-threadpool -c InstantMessaging_
home/config/iim.conf.xml id=muxin maxthreads=5
imconfutil -u add-server-threadpool -c InstantMessaging_
home/config/iim.conf.xml id=muxout maxthreads=5
imconfutil -u add-server-threadpool -c InstantMessaging_
home/config/iim.conf.xml id=serverin maxthreads=12
imconfutil -u add-server-threadpool -c InstantMessaging_
home/config/iim.conf.xml id=serverout maxthreads=12
```

2. Run the following commands to configure the service port.

The first command configures the service port for mux-to-server communications, and the second command configures the service port for server-to-server communications.

```
imconfutil -u set-listener-prop -c InstantMessaging_home/config/iim.conf.xml
c2s worker-in=muxin worker-out=muxout rcvbuf=1024000 sndbuf=1024000
imconfutil -u set-listener-prop -c InstantMessaging_home/config/iim.conf.xml
s2s worker-in=serverin worker-out=serverout rcvbuf=1024000 sndbuf=1024000
```

Creating Multiplexor Thread Pools

[Table 7–4](#) lists the properties that you configure to tailor the size and behavior of thread pools used by the multiplexor.

Table 7–4 Multiplexor Thread Pool Properties

Property	Description	Default Value
iim_mux.maxthreads	Maximum number of threads for the default thread pool.	10
iim_mux.threadpool.capacity	Capacity of the default thread pool.	-1 (no limit)
iim_mux.close_worker	Thread pool ID to clean up connections.	NA

You can configure thread pool size and behavior used to clean up socket connections. If this thread pool is not configured, the multiplexor uses the default thread pool. This thread pool, combined with the associated service ports, can improve multiplexor throughput.

To create multiplexor thread pools:

1. Run the following command to create the defined thread pool and to close sockets:

```
imconfutil -u -c InstantMessaging_home/config/iim.conf.xml add-mux-threadpool
id=closechannel maxthreads=5
```

2. Run the following command to assign the thread pool:

```
imconfutil -u -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
mux.close_worker=closechannel
```

Sample Load Test of the Instant Messaging Server

Table 7–5 shows a sample Instant Messaging Server load test on a server pool deployment.

Table 7–5 Sample Instant Messaging Server Load Test on a Server Pool Deployment

Platform Details	System Configuration	Server Heap Size	Number of Users	Number of Concurrent Sessions	User Cache	Load Per Second
<ul style="list-style-type: none"> Oracle VM Oracle Linux 6 OS 6v CPU (Server) 2v CPU (Multiplexor) RAM 32 Gbyte (server) RAM 8 Gbyte (multiplexor) 	Two servers and two multiplexors are installed on different hosts	6 GByte for multiplexor 22 Gbyte for server	110,000	110,000	256 count	<ul style="list-style-type: none"> 47 Users login to the server 47 Users logout 682 Presence updates 182 Chat messages to offline users 1357 Chat messages to online users 416 Chat messages to random users 520 Multiuser chat posts 47 Multiuser chat exit 47 Multiuser chat join 47 Multiuser change nickname 38 Roster additions 39 Roster removal 39 Roster rename

This sample uses the following configuration properties.

```
iim.jvm.maxmemorysize=22528
iim_mux.jvm.maxmemorysize=6144
iim_server.memory.user.cache_count=256
iim_ldap.maxconns=90
iim_server.maxthreads=100
iim_mux.maxthreads=15
iim_server.jvm.options=-d64
iim_mux.jvm.options=-d6
```

```
iim_mux.maxsessions=100000
iim_server.maxsessions=150000
```

where:

- **iim_server.memory.user.cache_count** specifies the memory user cache size. In this sample, the value is set to 256 for a user base of 1,10,000. If the user base is more than 1,10,000, increase this value proportionately.
- **iim_server.scratch_directory** specifies the directory where the user cache is written to the disk. Place the scratch directory on **tempfs**. For 1,10,000 user base, approximately 500 to 600 MBytes of space is required on a file system and approximately 4 to 5 GBytes of space is required on **tempfs**.
- **iim_ldap.maxconns** specifies the LDAP context pool size. In case of more roster operations and in a server pool environment, increase this value appropriately.
- **iim_server.maxthreads** specifies the size of the thread pool. If you do not have sufficient memory to keep user cache in **tempfs**, you can increase the value of the thread pool.
- **iim_server.jvm.options** enables use of the 64-bit JVM and thus large heap sizes.
- **iim_mux.jvm.options** enables you to start the multiplexor in 64-bit mode.
- **iim_mux.maxsessions** specifies the maximum number of concurrent client connections that a multiplexor can accept.
- **iim_server.maxsessions** specifies the number of sessions allowed through an instance of multiplexor connected to the server.

Controlling the Rate of Incoming Connections

You can control (throttle) the rate of incoming connections to both the Instant Messaging server and multiplexor.

- To control the rate of incoming connections to the Instant Messaging server, configure the **iim_server.c2s.tls.tokens** and **iim_server.c2s.tls.token_timeout** properties.
- To control the rate of incoming connections to the multiplexor, configure the **iim_mux.throttle.connections** and **iim_mux.throttle.timeout** properties.

[Table 7–6](#) describes the restricted properties you use to configure the rate of incoming connections to the Instant Messaging server and multiplexor:

Table 7–6 Incoming Connections Configuration Properties

Property	Default Value	Description
iim_server.c2s.tls.tokens	0	Specifies the number of connections allowed per timeout period. Use 0 (or a negative number) to disable.
iim_server.c2s.tls.token_timeout	200	Specifies timeout period in milliseconds.
iim_mux.throttle.connections	0	Specifies the number of connections allowed per timeout period. Use 0 (or a negative number) to disable.
iim_mux.throttle.timeout	1000	Specifies timeout period in milliseconds.

Tuning the XMPP WebSocket Gateway

Table 7–7 describes the properties you use to tune the XMPP WebSocket Gateway and multiplexor:

Table 7–7 XMPP WebSocket Gateway and Multiplexor Tuning Configuration Properties

Property	Default Value	Description
<code>iim_mux.close_worker</code>	NA	Thread pool used by the multiplexor for processing the close commands.
<code>iim_mux.maxthreads</code>	10	Maximum threads spawned by the multiplexor.
<code>iim_mux.maxsessions</code>	5000	Maximum number of client sessions accepted by the multiplexor.
<code>iim_websocket.threads.outgoing.poolsize</code>	10	Size of thread pool that handles outgoing packets from the XMPP WebSocket Gateway.
<code>iim_websocket.threads.translator.poolsize</code>	10	Size of thread pool that handles translation of packets.

Configuring Multiplexor Thread Pools

To configure multiplexor thread pools:

1. Run the following command to create the defined thread pool and to close sockets:

```
imconfutil -u -c InstantMessaging_home/config/iim.conf.xml add-mux-threadpool
id=closechannel maxthreads=5
```

2. Run the following command to assign the thread pool:

```
imconfutil -u -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
mux.close_worker=closechannel
```

Configuring Instant Messaging Server for High Availability

This chapter describes how to configure Oracle Communications Instant Messaging Server for high availability (HA).

Overview of High Availability for Instant Messaging Server

You can use server pooling to provide high availability (HA) for your Instant Messaging Server deployment. Server pools provide redundancy so that if one server in the pool fails, affected clients can reconnect and continue their sessions through another server in the pool with a minimum of inconvenience. Additionally, if you set up your deployment with load balancers, users can immediately reconnect and be directed by a load balancer to another node in the pool. You can also configure an Instant Messaging multiplexor with a list of Instant Messaging Server hosts for failover.

Note: In Instant Messaging Server versions prior to 10.0, Oracle Solaris Cluster was the recommended HA solution. As of Instant Messaging Server 10.0, Oracle Solaris Cluster is deprecated.

About Server Pooling

Server pooling enables you to support millions of users within a single domain. By using a server pool, you can share a domain across several servers in a server pool. In addition, you can use a load balancer to help manage server utilization in the pool.

By creating a server pool, the number of users you can support in an Instant Messaging Server deployment is no longer constrained by the capacity of a single server system. Instead, you can use the resources of several systems to support the users in a single domain. In addition, server pools provide redundancy so that if one server in the pool fails, affected clients can reconnect and continue their sessions through another server in the pool with a minimum of inconvenience. Deploying more than one server in a server pool creates a multi-node deployment.

You create a server pool by configuring the Instant Messaging servers to communicate over the server-to-server port and get user data from the same LDAP directory. Once you have configured the servers, you must configure the client resources to point to the load balancer, or load director, instead of a single node's host and port.

Caution: While it is possible to use a shared file system instead of an LDAP directory to store user properties, doing so negatively impacts performance and manageability. For this reason, only LDAP storage is supported for server pools.

To ensure that all servers within a server pool have consistent data, the following information is replicated among all servers in the pool:

- Routing information for end users
- Conference membership and configuration
- Multi-party conference messages

The following information is not replicated:

- One-on-one chat messages
- Presence subscriptions and notifications

If you are enforcing policy through access control files in your deployment, the content of the access control files must be the same among all servers in a server pool. See *Instant Messaging Server Security Guide* for more information.

Availability in an Instant Messaging Server Pool

If a node in a server pool goes down, all currently connected clients are disconnected and the sessions and resources become unavailable. If you set up your deployment with load balancers, users can immediately reconnect and be directed by a load balancer to another node in the pool. When they do so, they do not need to recreate conferences or news channels as this information is shared between servers in the pool. In addition, one-to-one chat sessions can be continued after the user is directed to another node in the pool.

Configuring Server-to-Server Communication Between Instant Messaging Servers

This section describes how to enable communication between two Instant Messaging servers, or peers, in a server pool. You must configure all servers in the pool with information about all other servers in the pool.

[Table 8–1](#) lists the configuration properties their values used to set up communication for two example Instant Messaging servers in a server pool, **iimA.siroe.com** and **iimB.siroe.com**.

See "[Configuration Properties](#)" for more information on the configuration properties.

Table 8–1 Example Configuration Information for Two Instant Messaging Servers in a Server Pool

Property	Value for Server A	Value for Server B	Notes
iim_server.serverid	iimA.siroe.com	iimB.siroe.com	In a server pool, this ID is used to support the dialback mechanism and is not used for authentication. This value should be unique within the server pool.
iim_server.password	secretforiimA	secret4iimB	None
iim_server.domainname	siroe.com	siroe.com	Peer servers within a server pool share the same default domain.

Note: When open federation is enabled, do not use the host name as the server ID. For example, the property `iim_server.serverid` should not be set to *host name*.

You define coserver properties by running the `imconfutil add-coserver` command. The `add-coserver` property enables you to set the server ID, the password used to authenticate for this coserver, the coserver host name, the domain server used by the coserver, and whether SSL is required.

After setting the `coserver` property, you can retrieve it by using the `imconfutil get-coserver-prop` command. If you need to modify an existing `coserver` property, use the `imconfutil set-coserver-prop` command. To remove a coserver, use the `imconfutil delete-coserver` command. If you need to verify the password of a coserver, use the `imconfutil verify-coserver-pass` command. To see a listing of all configured coservers, use the `imconfutil list-coservers` command.

See ["Setting Up Communication Between Two Instant Messaging Servers in a Server Pool"](#) for more information on coserver configuration.

Setting Up Communication Between Two Instant Messaging Servers in a Server Pool

The following example shows how to set up coservers `im1.example.com` and `im2.example.com`.

1. Perform the following commands on **host1 (im1.example.com)**.
 - a. Set the `iim_server.serverid` and `iim_server.password` configuration properties.


```
imconfutil set-prop -c InstantMessaging_home/config/iim.conf.xml iim_server.serverid=peer1.im1.example.com iim_server.password=peer1
```
 - b. Add the coserver (**im2.example.com**).


```
imconfutil add-coserver -c InstantMessaging_home/config/iim.conf.xml id=coserver1 serverid=peer2.im2.example.com password=peer2 host=im2.example.com domain=example.com
```
2. Perform the following commands on **host2 (im2.example.com)**.
 - a. Set the `iim_server.serverid` and `iim_server.password` configuration properties.


```
imconfutil set-prop -c InstantMessaging_home/config/iim.conf.xml iim_server.serverid=peer2.im2.example.com iim_server.password=peer2
```
 - b. Add the coserver (**im1.example.com**).


```
imconfutil add-coserver -c InstantMessaging_home/config/iim.conf.xml id=coserver1 serverid=peer1.im1.example.com password=peer1 host=im1.example.com domain=example.com
```
3. Restart Instant Messaging Server on both hosts.


```
imadmin refresh server
```

Adding a New Node to an Existing Instant Messaging Server Deployment

If you need to add an additional node to an existing server pool, you need must configure the new server for server-to-server communication then add configuration

information about the new server to all existing servers in the pool. In addition, you must add configuration information about all the servers in the pool to the new node. See "[Setting Up Communication Between Two Instant Messaging Servers in a Server Pool](#)" for instructions.

Securing a Multi-node Deployment

When a node connects to a remote server, the node provides a *dialback* key. The remote server then connects back to the node in order to verify the dialback key. In a multi-node deployment, the remote server may connect back to a different node in the pool from the node that originally sent the dialback key. The node the remote server connects to must provide the same dialback key that the original connecting node supplied. The `iim_server.dialback.key` configuration property defines which dialback key a node should use. The value for the dialback key is randomly generated unless you explicitly specify one. See "[Manually Defining the Dialback Key for an Instant Messaging Server in a Server Pool](#)" for instructions.

The **From** attribute is used by a remote server to connect back to an initiating server. Typically, a server's domain name is used as the value for the **From** attribute in server-to-server communication under Jabber. However, all servers in a server pool share the same domain name. Therefore, the domain name cannot be used as a key to locate a single server in a pool. Instead, Instant Messaging Server uses a server or peer identifier (*serverid*) instead of the domain name as the value for the **From** attribute.

Manually Defining the Dialback Key for an Instant Messaging Server in a Server Pool

The value for the dialback key is randomly generated unless you explicitly specify one.

1. Use the `imconfutil` command to modify the value of the `iim_server.dialback.key` configuration property.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml iim_
server.dialback.key=mymultinodedialbackkey
```

2. Refresh the configuration on both servers.

```
imadmin refresh server
```

Using Shoal for Server Pool Messaging

Instant Messaging Server uses Shoal, a Java technology-based scalable and dynamic clustering framework to connect multiple servers within a server pool. For more information on Shoal, see the Project Shoal website at:

<https://shoal.java.net>

Setting Shoal Properties

To enable Shoal, use the `imconfutil` command to set the following configuration properties:

- `iim_server.serverid=servername` (Ensure that this value is unique for each server)
- `iim_server.password=password` (Ensure that this password is same across all servers)

For example:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
```

```
server.serverid=server1 iim_server.password=password
```

Using Shoal for Automatic Discovery of Peer Servers in a Pool

Instant Messaging Server enables you to use the Shoal clustering framework to automatically discover and add peer servers in a server pool. The following steps describe how to configure Shoal for the servers in a pool that belong to the same IP subnet. To configure Shoal for servers in a pool that are part of different subnets, see ["Using Shoal Across Subnets"](#).

To enable auto-discovery of peer servers:

1. Configure a server pool containing a number of Instant Messaging servers to use the LDAP **propstore** property.
2. Use the **imconfutil** command to set the following configuration property to start auto-discovery.

```
imconfutil -c InstantMessaging_home/iim.conf.xml set-prop iim_
server.peer.autodiscover=true
```

3. Set the configuration properties as explained in ["Setting Shoal Properties"](#).

Setting the properties enables you to start and stop the servers as required. If you are connected to one server, you can see the presence of the server and chat with users on any other server.

Using Shoal for Conferences Across Server Pools

Instant Messaging Server enables the use of Shoal group messaging to broadcast conference messages across the server pool. Shoal framework can be used to send conference messages across the server pool even if you have not used Shoal for auto-discovery or across subnets. When you enable use of Shoal across server pools, all conference presence broadcasts including join and leave notifications, messages, and chat status notifications will be sent using the Shoal group messaging feature.

To enable Shoal for conferences:

1. Set the properties as explained in ["Setting Shoal Properties"](#).
2. Use the **imconfutil** command to set the following configuration property.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.peer.conferences.usep2p=true
```

This property is used to enable or disable the use of Shoal for conference messaging. If you set the property to false or not set at all, the legacy server-to-server connection is used.

You can enable Shoal anytime during and after configuration. If you enable this feature after configuration, restart all the servers.

Note: When using Shoal for peer discovery and conferences, ensure that:

- The **iim_server.password** property is the same on all hosts.
 - Relay is enabled for communication to work when hosts are on different subnets.
-
-

Using Shoal Across Subnets

The Shoal configuration of a server pool in a subnet cannot discover new peers that are present in different IP subnets. Shoal uses relay nodes to propagate peer information across subnets. You must configure Instant Messaging Server to start a separate process that performs the Shoal relay functionality, by providing connection details of the relays present in different subnets.

To enable Shoal across different subnets, you must start the relay server. To start the relay server, you need at least one relay server per subnet. You can configure any number of relay servers.

To start the relay server, use the **imconfutil** command to set the **relay.imadmin.enable** and **relay.listen_address** (optional) configuration properties. For example:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop
relay.imadmin.enable=true relay.listen_address=192.0.2.0
```

The list of relay servers is specified by using the **relay.uri_list** property:

```
relay.uri_list = list of relays
```

You specify each relay by using a URI of the form **tcp://host:port**. For example:

```
relay.uri_list = tcp://relay2.example.com:5600, tcp://relay3.example.com:5600
```

You can start or stop the relay process independently of the Instant Messaging server. Stopping or restarting the relay process does not affect the servers that are already in the pool.

About Multiplexor Failover

You can configure an Instant Messaging multiplexor with a list of Instant Messaging Server hosts for failover. When the multiplexor detects a failure with the current Instant Messaging Server host, it attempts a connection with another Instant Messaging Server host in the configured list. After the multiplexor connects to the next Instant Messaging Server host in the list, it informs incoming clients of the newly connected Instant Messaging Server host. During the failover process, the multiplexor closes its listener port for incoming clients (default 5222) until a new Instant Messaging Server host is available. When the multiplexor encounters a failure with an Instant Messaging Server host, it retries the next host in the list for a specified interval before it tries the next available host.

You can configure multiplexor failover in either *polling* or *non-polling* mode.

Polling mode works as follows:

1. The multiplexor attempts to connect to the first Instant Messaging server in the failover list.
2. If this first server does not respond, the multiplexor keeps polling it until the server connects back.
3. In the meantime, the multiplexor fails over to the other available Instant Messaging servers in the failover list, and handles the client connection with the newly connected server.
4. If the first Instant Messaging server becomes available again, then the multiplexor starts handling new incoming clients with the server that was first polled.
5. The multiplexor maintains connection to the failed-over Instant Messaging server as long as it has connected clients.

In polling mode, if the original Instant Messaging Server host becomes available again, then the multiplexor starts handling new incoming clients with the polled first server. The failed-over Instant Messaging Server host is kept connected as long as it has connected clients.

In non-polling mode, when an Instant Messaging server fails, the multiplexor tries the list of Instant Messaging servers in round-robin fashion and connects to the first available server.

The default polling interval, specified by the `iim_mux.polling_interval` property, is five seconds. A positive value means that the multiplexor operates in polling mode. Thus, the default is polling mode. A negative polling interval means that the multiplexor operates in non-polling mode.

Enabling Multiplexor Failover

To enable the multiplexor to fail over to other Instant Messaging Server hosts, configure the following properties:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_mux.polling_
interval=interval
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
mux.serverport=ims_host1:listener_port,ims_host2:listener_port
```

where:

- *interval* is the polling interval, in seconds, during which the multiplexor attempts to contact the next Instant Messaging Server for failover
- *ims_host1* is the first Instant Messaging Server host for failover, *ims_host2* is the next, and so on
- *listener_port* is the port on which the Instant Messaging Server host communicates with the multiplexor (default is 45222)

Overview of Using Oracle Solaris Cluster

This section provides information about Oracle Solaris Cluster HA requirements, the terms used in examples, and the permissions that you need to configure HA.

Note: As of Instant Messaging Server 10.0, Oracle Solaris Cluster has been deprecated.

HA Configuration Software Requirements

[Table 8–2](#) shows the required software for an Instant Messaging Server HA deployment.

Table 8–2 HA Software Requirements

Software and Version	Notes and Patches
Oracle Solaris 10	All versions of Oracle Solaris 10 are supported. Oracle Solaris 10 requires at least Oracle Solaris Cluster 3.0 Update 3. Oracle Solaris 10 includes Oracle Solaris Logical Volume Manager (LVM).

Table 8–2 (Cont.) HA Software Requirements

Software and Version	Notes and Patches
Oracle Solaris Cluster 3.1 or 3.2	<p>Oracle Solaris Cluster software must be installed and configured on all the nodes in the cluster. To install Oracle Solaris Cluster 3.1 or 3.2, use the Sun Java Enterprise System installer by following the installation process in <i>Sun Java Enterprise System 5 Update 1 Installation Guide for UNIX</i> at:</p> <p>http://docs.oracle.com/cd/E19528-01/820-2827/index.html</p> <p>After you install the Oracle Solaris Cluster software, you must configure the cluster. For more information, see <i>Sun Cluster System Administration Guide for Solaris OS</i> at:</p> <p>http://docs.oracle.com/cd/E19787-01/819-2971</p> <p>Oracle Solaris Cluster Patches - For Oracle Solaris 10, you can download patches from My Oracle Support at:</p> <p>https://support.oracle.com</p>
Oracle Solaris Volume Manager	Oracle Solaris 10.
Veritas Volume Manager (VxVM)	Oracle Solaris 10 requires at least version 3.5 and the required patches.
Veritas File System (VxFS)	Oracle Solaris 10 requires at least version 3.5 and the required patches.

HA Configuration Requirements

To install and configure an Instant Messaging Server HA configuration, log in or become **root** and specify a console or window for viewing messages that exist in the `/dev/console` directory.

HA Configuration Terms and Checklist

[Table 8–3](#) describes the variables used in the configuration examples in this chapter. In addition, you must gather the information before you configure HA for Instant Messaging Server. You are prompted for this information during configuration. Use this checklist along with the system requirements specified in *Instant Messaging Server Installation and Configuration Guide*.

Table 8–3 Configuration Examples Variables

Name in Example	Description
<code>/global/im</code>	Global file system or cluster file system (CFS) mount point.
<code>/local/im</code>	FFS mount point for the shared disk.
<code>LOG_HOST_RS</code>	Logical host name resource.
<code>IM_NODE1</code>	Node1 of the cluster.
<code>IM_NODE2</code>	Node2 of the cluster.
<code>IM_RG</code>	Instant Messaging Server resource group.
<code>IM_HASP_RS</code>	Instant Messaging Server storage resource.
<code>IM_SVR_RS</code>	Instant Messaging Server resource.
<code>IM_RUNTIME_DIR</code>	Either global or FFS mount point. The value is <code>/global/im</code> or <code>/local/im</code> .

Table 8–3 (Cont.) Configuration Examples Variables

Name in Example	Description
IM_SVR_BASE	Instant Messaging Server base installation directory. The default value is <code>/opt/sun/comms/im</code> .
IM_SCHA_BASE	Instant Messaging Server HA agent base installation directory. The default value is <code>/opt/sun/comms/im_scha</code> .
IM_RUNTIME_CONFIG	Location of the Instant Messaging Server runtime directory <i>InstantMessaging_runtime/default/config</i> .
INSTALL-ROOTIM1	Installation directory for instance 1 in a symmetric setup. For example <code>/opt/node1</code> .
INSTALL-ROOTIM2	Installation directory for instance 2 in a symmetric setup. For example <code>/opt/node2</code> .

Starting and Stopping the Instant Messaging Server HA Service

To start and stop the Instant Messaging Server HA service, use the Oracle Solaris Cluster **scswitch** command.

Caution: Do not use the **imadmin start**, **imadmin stop**, or **imadmin refresh** commands in a HA environment with Sun Cluster. Instead, use the Oracle Solaris Cluster administrative utilities. For more information about the Oracle Solaris Cluster **scswitch** command, see *Oracle Solaris Cluster Reference Manual*.

To start the Instant Messaging Server HA service, enter the following command:

```
scswitch -e -j IM_SVR_RS
```

To stop the Instant Messaging Server HA service, enter the following command:

```
scswitch -n -j IM_SVR_RS
```

To restart the Instant Messaging Server HA Service, enter the following command:

```
scswitch -R -j IM_SVR_RS
```

Troubleshooting the Instant Messaging Server HA Configuration

Troubleshooting error messages are stored in the error log. The logs are controlled by the **syslog** facility. For information about using the logging facility, see the **syslog.conf** man page.

Setting Up HA for Instant Messaging Server

This section describes the steps to set up HA for Instant Messaging Server.

Choosing a High Availability Model for Your Instant Messaging Server Deployment

This section lists the HA models, and describes the procedure to install and configure the asymmetric and symmetric models for deployment.

[Table 8–4](#) summarizes the advantages and disadvantages of each HA model. Use this information to decide the appropriate model for your deployment.

Table 8–4 HA Models Advantages and Disadvantages

Model	Advantages	Disadvantages	Recommended Users
Asymmetric	<ul style="list-style-type: none"> Simple Configuration Backup node is 100% reserved. Rolling upgrade with negligible downtime 	<ul style="list-style-type: none"> Machine resources are not fully utilized. 	<ul style="list-style-type: none"> A small service provider with plans to expand in the future.
Symmetric	<ul style="list-style-type: none"> Efficient use of system resources Higher availability 	<ul style="list-style-type: none"> Resource contention on the backup node. HA requires fully redundant disks. 	<ul style="list-style-type: none"> A small corporate deployment that can accept performance penalties if a single server fails.
N+1	<ul style="list-style-type: none"> Load distribution Easy expansion 	<ul style="list-style-type: none"> Management and configuration complexity. 	<ul style="list-style-type: none"> A large service provider who requires distribution with no resource constraints.

High-Level Task List for an Asymmetric HA Deployment

The following is a list of the tasks necessary to install and configure Instant Messaging Server for asymmetric HA:

1. Prepare the nodes.
 - a. Install the Oracle Solaris operating system on all the nodes of the cluster.
 - b. Install Oracle Solaris Cluster software on all the nodes of the cluster.
 - c. Install the Instant Messaging Server HA Agents package, **SUNWiimsc**, on all the nodes of the cluster by using the Installer.
 - d. Create a file system on the shared disk.
 - e. Install Instant Messaging Server on all the nodes of the cluster by using the Installer.
 - f. Create a symbolic link from the Instant Messaging Server `/etc/opt/sun/comms/im` directory to the shared disk `InstantMessaging_runtime` directory on all the nodes of the cluster.
2. Configure the first or the primary node.
 - a. Using the Oracle Solaris Cluster command-line interface, set up HA on the primary node.
 - b. Run the Instant Messaging Server **configure** utility on the primary node.
 - c. Using the Oracle Solaris Cluster command-line interface, create and enable a resource group for Instant Messaging Server.

See "[Installing and Configuring in an Asymmetric HA Environment](#)" for step-by-step instructions.

High-Level Task List for a Symmetric HA Deployment

The following is a list of the tasks necessary to install and configure Instant Messaging Server for symmetric HA:

1. Prepare the nodes.
 - a. Install the Oracle Solaris operating system software on all the nodes of the cluster.

- b. Install the Oracle Solaris Cluster software on all the nodes of the cluster.
 - c. Create four file systems. You can create a CFS or global file systems or FFS' or local file systems.
 - d. Create the necessary directories.
 - e. Install the Instant Messaging Server HA Agents package, **SUNWiimsc**, on all nodes of the cluster by using the Installer.
2. Install and configure the first instance of Instant Messaging Server HA.
 - a. Using the Installer, install Instant Messaging Server on the first node of the cluster.
 - b. Using the Oracle Solaris Cluster command-line interface, configure HA on the first node.
 - c. Create a symbolic link from the Instant Messaging Server `/etc/opt/sun/comms/im` directory to the shared disk *InstantMessaging_runtime* directory on the first node.
 - d. Run the Instant Messaging Server **configure** utility on the first node.
 - e. Using the Oracle Solaris Cluster command-line interface, create and enable a resource group for Instant Messaging Server on the first node.
 - f. Using the Oracle Solaris Cluster command-line interface to test the successful creation of the resource group, perform a failover to the second node.
3. Install and configure the second instance of Instant Messaging Server HA.
 - a. Using the Installer, install Instant Messaging Server on the second node of the cluster.
 - b. Using the Oracle Solaris Cluster command-line interface, configure HA on the second node.
 - c. Create a symbolic link from the Instant Messaging Server `/etc/opt/sun/comms/im` directory to the shared disk *InstantMessaging_runtime* directory on the secondary node.
 - d. Run the Instant Messaging Server **configure** utility on the second node.
 - e. Using the Oracle Solaris Cluster command-line interface, create and enable a resource group for Instant Messaging Server on the second node.
 - f. Using the Oracle Solaris Cluster command-line interface to test the successful creation of the resource group, perform a failover to the first node.

See ["Installing and Configuring in a Symmetric HA Environment"](#) for step-by-step instructions.

Installing and Configuring in an Asymmetric HA Environment

This section contains instructions for configuring an asymmetric HA Instant Messaging Server cluster.

Creating File Systems for HA Deployment

Create a file system on the shared disk. The `/etc/vfstab` directory should be identical on all the nodes of the cluster.

For the CFS, the directory should be similar to the following example.

```
## Cluster File System/Global File System ##
```

```
/dev/md/penguin/dsk/d400 /dev/md/penguin/rdisk/d400 /global/im ufs 2 yes
global,logging
```

For the failover FFS, the directory should be similar to the following example.

```
## Fail Over File System/Local File System ##
/dev/md/penguin/dsk/d400 /dev/md/penguin/rdisk/d400 /local/im ufs 2 no logging
```

Note: The fields in these commands are separated by tabs and not spaces.

Creating the Instant Messaging Server Directory on all the Shared Disks of the Cluster in the HA Deployment

For all the nodes of the cluster, create a directory, *InstantMessaging_runtime*, to store the configuration details and data. For example, to create an Instant Messaging Server directory on a shared disk, enter either one of the following:

```
mkdir -p /local/im
```

or

```
mkdir -p /global/im
```

Installing and Configuring HA for Instant Messaging Server Software

This section contains instructions for the tasks involved in installing and configuring HA for Instant Messaging Server. Perform the following tasks to complete the configuration:

- Preparing Each Node of the Cluster
- Setting Up the Primary Node
- Invoking the configure Utility on the Primary Node

Preparing Each Node of the Cluster

For each node in the cluster, create the Instant Messaging Server runtime user and group under to run the components. The user ID (UID) and group ID (GID) numbers must be the same on all the nodes in the cluster.

- Runtime UID: User name using which the Instant Messaging server runs. The default value is **inetuser**.
- Runtime GID: Group using which the Instant Messaging server runs. The default value is **inetgroup**. Although the **configure** utility creates the IDs, you can create the IDs before you invoke the **configure** utility as part of the preparation of each node. Create the runtime UID and GID on a node where you will not invoke the **configure** utility, which is usually secondary node.

Ensure that the user name, group name and the corresponding UID and GID are the same in the following files on all nodes:

- **inetuser** or the name that you select in the **/etc/passwd** directory on all the nodes in the cluster
- **inetgroup** or the name that you select in the **/etc/group** directory on all the nodes in the cluster

Refer to your operating system documentation for detailed information about users and groups.

Selecting the Default Installation Directory "IM_SCHA"

For Instant Messaging Server and Instant Messaging Server Oracle Solaris Cluster agent **IM_SCHA**, the Installer uses the **/opt/sun/comms** directory on the Oracle Solaris operating system as the default installation directory. The value of the *InstantMessaging_home* variable is **/opt/sun/comms/im**.

However, if you are using a shared disk for binaries, you must specify a CFS or a FFS installation directory. For example, if **/global/im/** is the installation directory, then the value of *InstantMessaging_home* is **/global/im/im**.

If you are using a local disk, you should install the Instant Messaging in the same directory on each machine in the node.

- Configuration files and runtime files reside on a CFS or on a highly-available FFS. Binaries are installed on local file systems on each node at the same location. Enables rolling upgrade of the Instant Messaging Server software.
- Binaries, configuration files and runtime files either reside on a CFS or on a highly-available FFS. The Instant Messaging Server installation is required only on one node as the binaries are shared across all the nodes. Instant Messaging Server upgrade needs a server down time.

Installing Instant Messaging Server Products and Packages

Install products and packages by using the Installer. For more information about the installer, see *Unified Communications Suite Installation and Configuration Guide*.

[Table 8–5](#) lists the products or packages required for a multiple node cluster configuration.

Table 8–5 Requirements for Multiple Nodes

Product or Package	Node 1	Node n
Oracle Solaris Cluster Software	Yes	Yes
Instant Messaging Server Server	Yes	Yes, if you use a local disk for configuration files and binaries. No, if you use a shared disk for configuration files and binaries.
Oracle Solaris Cluster Agent for Instant Messaging Server SUNWiimsc	Yes	Yes, if you use a local disk for configuration files and binaries. No, if you use a shared disk for configuration files and binaries.
Shared components	Yes	Yes

Instant Messaging Server HA Agent Installation

To install the Instant Messaging Server Oracle Solaris Cluster HA agent:

1. Run the Installer in the global zone:

```
commpkg install
```

On Solaris 10 zones, run the **commpkg** command from global and non-global zones.

2. Select the Instant Messaging Server Oracle Solaris Cluster HA Agent software when prompted.
3. Enter the Oracle Solaris Cluster HA Agent preconfiguration command.

```
IM_SCHA_BASE/bin/init-config
```

On Solaris 10 zones, run this command only from the global zone.

Setting Up the Primary Node

Use the Oracle Solaris Cluster command line interface to set up HA on the first node.

1. Register the Instant Messaging Server and **HAStoragePlus** resource.

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

2. Create a failover Instant Messaging Server resource group. For example, for a two node asymmetric cluster setup, the following command creates the Instant messaging resource group **IM-RG** with the primary node as **NODE1** and the secondary, or failover, node as **NODE2**.

```
scrgadm -a -g IM-RG -h IM_NODE1,IM_NODE2
```

3. Create a logical hostname resource in the Instant Messaging Server resource group and change the resource group state to online. For example, the following instructions create the logical hostname resource **LOG_HOST_RS** and bring the resource group **IM-RG** to online state.

```
scrgadm -a -L -g IM-RG -l LOG_HOST_RS
scrgadm -c -j LOG_HOST_RS -y \
R_description="LogicalHostname resource for LOG_HOST_RS"
scswitch -Z -g IM-RG
```

4. Create and enable the **HAStoragePlus** resource. For example, the following commands create and enable the **HAStoragePlus** resource **IM_HASP_RS**.

```
scrgadm -a -j IM_HASP_RS -g IM-RG -t
SUNW.HAStoragePlus:4 -x FilesystemMountPoints=/IM_RUNTIME_DIR
scrgadm -c -j IM_HASP_RS -y
R_description="Failover data service resource for SUNW.HAStoragePlus:4"
scswitch -e -j IM_HASP_RS
```

5. Create a symbolic link from the Instant Messaging Server **/etc/opt/sun/comms/im** directory to the shared disk *InstantMessaging_runtime* directory on all the nodes of the cluster.

For example, enter the following commands on all the nodes of the cluster:

```
cd /etc/opt/sun/comms
ln -s /IM_RUNTIME_DIR im
```

Invoking the configure Utility on the Primary Node

1. Invoke the **configure** utility.

For example, from the *InstantMessaging_home* directory enter the following command:

```
# pwd
/IM_SVR_BASE
# ./configure
```

For more information about the **configure** utility, see *Instant Messaging Server Installation and Configuration Guide*.

2. When prompted for the Instant Messaging Server runtime files directory *InstantMessaging_runtime*, enter either of the following commands:

- a. If you are using FFS for the runtime files, enter **/local/im**.
- b. If you are using a CFS for the runtime files, enter **/global/im**.

3. If prompted for the Instant Messaging Server host name, enter the logical host. Choose to accept the logical host even if the **configure** utility is unable to connect to the specified host. The logical host resource might be offline at the time when you invoke the **configure** utility.
4. Do not start Instant Messaging Server after configuration or on system startup.
5. Copy the Instant Messaging Server configuration file **iim.conf.xml** to the **iim.conf** file with the same permissions.

Note: Also copy the **iim.conf.xml** file to **iim.conf** after any future configuration changes as cluster uses the **iim.conf** file.

6. To use the Gateway Connector service in HA, update this service configuration with the virtual host name or IP address and port number as follows:

```
imconfutil --config config_file_path iim_gwc.hostport=virtual host-name or ip:port
```

For example:

```
/opt/sun/comms/sbin/imconfutil --config /DATA1/default/config/iim.conf.xml iim_gwc.hostport=192.10.12.11:22222
```

7. Create and enable the Instant Messaging Server resource.

In this example, the resource group name is **IM_SVR_RS**. Provide the logical host resource name and the **HASStoragePlus** resource name. For example,

```
scrgadm -a -j IM_SVR_RS -g IM-RG
-t SUNW.iim -x Server_root=/InstantMessaging_home
-x Confdir_list=/InstantMessaging_runtime (ex: /local/im/default/config )
-y Resource_dependencies=IM_HASP_RS,LOG_HOST_RS
scrgadm -e -j IM_SVR_RS
```

8. Test the successful creation of the Instant messaging resource group by performing a failover.

```
scswitch -z -g IM-RG -h IM_NODE2
```

Note: You do not need to configure the second node as the configuration is shared between all the nodes by soft links pointing to the shared location.

Installing and Configuring in a Symmetric HA Environment

This section contains instructions for configuring a symmetric HA Instant Messaging Server system.

Initial Tasks

You must complete the following preparatory tasks before installing Instant Messaging Server on the nodes. The preparatory tasks are:

- Creating File Systems
- Installing the Instant Messaging Server HA Package
- Preparing Each Node of the Cluster

Creating File Systems

Instant Messaging Server binaries, configuration files, and runtime files reside on the CFS or on the highly available FFS. For each Instant Messaging Server instance, installation is needed on only one node as the binaries are shared across all the nodes.

To create file systems:

1. Create four file systems by using CFS or FFS.

To create a system by using CFS, for example, the contents of the `/etc/vfstab` file should appear as follows.

```
# Cluster File System/Global File System ##
/dev/md/penguin/dsk/d500 /dev/md/penguin/rdisk/d500
/INSTALL-ROOTIM1 ufs 2 yes logging,global
/dev/md/penguin/dsk/d400 /dev/md/penguin/rdisk/d400
/share-disk-dirIM1 ufs 2 yes logging,global
/dev/md/polarbear/dsk/d200 /dev/md/polarbear/rdisk/d200
/INSTALL-ROOTIM2 ufs 2 yes logging,global
/dev/md/polarbear/dsk/d300 /dev/md/polarbear/rdisk/d300
/share-disk-dirIM2 ufs 2 yes logging,global
```

Note: The fields must be separated by tabs.

To create a system by using FFS, for example, the contents of the `/etc/vfstab` file should appear as follows.

```
# Failover File System/Local File System ##
/dev/md/penguin/dsk/d500 /dev/md/penguin/rdisk/d500
/INSTALL-ROOTIM1 ufs 2 yes logging
/dev/md/penguin/dsk/d400 /dev/md/penguin/rdisk/d400
/share-disk-dirIM1 ufs 2 yes logging
/dev/md/polarbear/dsk/d200 /dev/md/polarbear/rdisk/d200
/INSTALL-ROOTIM2 ufs 2 yes logging
/dev/md/polarbear/dsk/d300 /dev/md/polarbear/rdisk/d300
/share-disk-dirIM2 ufs 2 yes logging
```

Note: The fields must be separated by tabs.

2. Create the following mandatory directories on all the nodes of the cluster.

```
# mkdir -p /INSTALL-ROOTIM1 share-disk-dirIM1
INSTALL-ROOTIM2 share-disk-dirIM2
```

Installing the Instant Messaging Server HA Package

Install the Instant Messaging Server Oracle Solaris Cluster HA package in two nodes. You can use the Communication Suite 7 Update 2 installer to install the HA package.

To install the Instant Messaging Server Oracle Solaris Cluster HA agent:

1. Run the Installer:

```
commpkg install
```

In Solaris 10 zones, run this command from the global and non-global zones.

2. When prompted, select the Instant Messaging Server Oracle Solaris Cluster HA Agent software.
3. Run the Sun Cluster HA Agent pre-configuration command:

`IM_SCHA_BASE/bin/init-config`

On Solaris 10 zones, run this command only from the global zone.

Preparing Each Node of the Cluster

For each node in the cluster, create the Instant Messaging Server runtime user and group under which the components will run. The UID and GID numbers must be the same on all nodes in the cluster.

- Runtime UID: User name using which the Instant Messaging server runs. The default value is **inetuser**.
- Runtime GID: Group using which the Instant Messaging server runs. The default value is **inetgroup**. Although the **configure** utility creates these IDs, you can create the IDs before you invoke the **configure** utility as part of the preparation of each node. Create the runtime UID and GID on a node where you might not invoke the **configure** utility, which is usually secondary node.

Ensure that the user name, group name and the corresponding UID and GID are same in the following files on all nodes:

- **inetuser** or the name that you select in the `/etc/passwd` directory on all the nodes in the cluster
- **inetgroup** or the name that you select in the `/etc/group` directory on all the nodes in the cluster

Refer to your operating system documentation for detailed information about users and groups.

Installing and Configuring the First Instance of Instant Messaging Server

To install the first instance of Instant Messaging Server:

1. Verify whether the files are mounted.

On the primary node Node1, enter the following command:

```
df -k
```

The following message shows a sample output:

```
/dev/md/penguin/dsk/d500 35020572
34738 34635629 1% /INSTALL-ROOTIM1
/dev/md/penguin/dsk/d400 35020572
34738 34635629 1% /share-disk-dirIM1
```

2. Using the Installer, install Instant Messaging Server on the primary node.

- a. Run the Installer:

```
commpkg install
```

Note: In case of Oracle Solaris 10 zones, refer to *Unified Communications Suite Installation and Configuration Guide*.

- b. At the Specify Installation Directories prompt, enter the installation root **INSTALL-ROOTIM1**.
3. Create a symbolic link from the Instant Messaging Server the `/etc/opt/sun/comms/im` directory to the shared disk `IM_RUNTIME_DIR` directory

on all the nodes of the cluster. For example, enter the following commands on a cluster node:

```
# cd /etc/opt/sun/comms
# ln -s /share-disk-dirIM1 im
```

To configure Oracle Solaris Cluster on the first node by using the Oracle Solaris Cluster command-line interface:

1. Register the following resource types.

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

2. Create a failover resource group.

In the following example, the resource group is **IM-RG1**, **IM_NODE1** is the primary node and **IM_NODE2** is the failover node.

```
scrgadm -a -g IM-RG1 -h IM_NODE1,IM_NODE2
```

3. Create a logical host name resource for the node.

Add the logical host name **LOG_HOST_RS** to the resource group. Instant Messaging Server listens on this host. The following example uses **LOG-HOST-IM-RS1**. Replace this value with the actual hostname.

```
scrgadm -a -L -g IM-RG1 -l LOG-HOST-IM-RS1
scrgadm -c -j LOG-HOST-IM-RS1 -y R_description=
"LogicalHostname resource for LOG-HOST-IM-RS1"
```

4. Bring the resource group online.

```
scswitch -Z -g IM-RG1
```

5. Create a **HAStoragePlus** resource and add it to the failover resource group.

In this example, the resource is called **IM_HASP_RS1**. Replace the resource with your own resource name.

Note: The example is split for display purpose in this document.

```
scrgadm -a -j IM-HASP-RS1 -g IM-RG1 -t
SUNW.HAStoragePlus:4 -x FilesystemMountPoints=/INSTALL-ROOTIM1,
/share-disk-dirIM1
scrgadm -c -j IM-HASP-RS1 -y R_description="Failover data
service resource for SUNW.HAStoragePlus:4"
```

6. Enable the **HAStoragePlus** resource.

```
scswitch -e -j IM-HASP-RS1
```

To configure the first instance of Instant Messaging Server:

1. Run the **configure** utility on the primary node.

```
# cd INSTALL-ROOTIM1/im
# ./configure
```

For more information about the **configure** utility, see *Instant Messaging Server Installation and Configuration Guide*.

2. When prompted for the Instant Messaging Server Runtime Files Directory, enter **/share-disk-dirIM1** if you are using **HAStoragePlus** for the runtime files.
3. When prompted for the Instant Messaging Server host name, enter the logical host.

Choose to accept the logical host even if the **configure** utility cannot connect to the specified host. The logical host resource might be offline at the time when you invoke the **configure** utility.

4. Do not start Instant Messaging Server after configuration or on system startup.
5. Copy the Instant Messaging Server configuration file **iim.conf.xml** to the **iim.conf** file with the same permissions.

Note: Also copy the **iim.conf.xml** file to **iim.conf** after any future configuration changes as cluster uses the **iim.conf** file.

6. To use the Gateway Connector service in HA, update this service configuration with the virtual host name or IP address and port number as follows:

```
InstantMessaging_home/imconfutil --config config_file_path iim_
gwc.hostport=virtual host-name or ip:port
```

For example:

```
/opt/sun/comms/sbin/imconfutil --config /DATA1/default/config/iim.conf.xml iim_
gwc.hostport=192.10.12.11:22222
```

7. Create and enable the Instant Messaging Server resource.

In this example, the resource group name is **IM_SVR_RS1**. Provide the logical host resource name and the **HAStoragePlus** resource name.

```
scrgadm -a -j IM_SVR_RS1 -g IM-RG1
-t SUNW.iim -x Server_root=/INSTALL-ROOTIM1/im
-x Confdir_list=/share-disk-dirIM1/default/config
-y Resource_dependencies=IM-HASP-RS1,LOG-HOST-IM-RS1
scrgadm -e -j IM_SVR_RS1
```

8. Test the successful creation of the Instant Messaging Server resource group by performing a failover.

```
scswitch -z -g IM-RG1 -h IM_NODE2
```

Note: You do not have to configure the second node as configuration is shared between all the nodes by soft links pointing to shared location.

Installing and Configuring the Second Instance of Instant Messaging Server

To install the second instance of Instant Messaging Server:

1. Verify whether the files are mounted. On the primary node **IM_NODE2**, enter:

```
df -k
```

The following output is displayed:

```
/dev/md/polarbear/dsk/d300 35020572
34738 34635629 1% /share-disk-dirIM2
```

```
/dev/md/polarbear/dsk/d200 35020572
34738 34635629 1% /INSTALL-ROOTIM2
```

2. Install Instant Messaging Server on the primary node.
 - a. Run the Installer:

```
commpkg install
```
 - b. At the Specify Installation Directories prompt, specify the installation root **INSTALL-ROOTIM2**.
3. Create a symbolic link from the Instant Messaging Server **/etc/opt/sun/comms/im** directory to the shared disk **IM_RUNTIME_DIR** directory on this cluster node.

For example, enter the following commands on all the nodes of the cluster:

```
cd /etc/opt/sun/comms
ln -s /share-disk-dirIM2 im
```

Configuring Oracle Solaris Cluster on the Second Node

To configure Oracle Solaris Cluster on the second node by using the Oracle Solaris Cluster command-line interface:

1. Create a failover resource group.

In the following example, the resource group is **IM-RG2**, **IM_NODE2** is the primary node and **IM_NODE1** is the failover node.

```
scrgadm -a -g IM-RG2 -h IM_NODE2,IM_NODE1
```

2. Create a logical host name resource for this node.

Add the logical host name **LOG_HOST_RS** to the resource group. Instant Messaging Server listens on this host. The following example uses **LOG-HOST-IM-RS2** in the place where you will substitute in the actual host name.

```
scrgadm -a -L -g IM-RG2 -l LOG-HOST-IM-RS2
scrgadm -c -j LOG-HOST-IM-RS2 -y R_description=
"LogicalHostname resource for LOG-HOST-IM-RS2"
```

3. Bring the resource group online.

```
scswitch -Z -g IM-RG2
```

4. Create a **HASStoragePlus** resource and add it to the failover resource group.

In this example, the resource is called **IM-HASP-RS2**. Replace it by your own resource name. The lines are divided and show as two lines in the example for display purposes in this document.

```
scrgadm -a -j IM-HASP-RS2 -g IM-RG2 -t
SUNW.HASStoragePlus:4 -x FilesystemMountPoints=/INSTALL-ROOTIM2,
/share-disk-dirIM2
scrgadm -c -j IM-HASP-RS2 -y R_description="Failover data
service resource for SUNW.HASStoragePlus:4"
```

5. Enable the **HASStoragePlus** resource.

```
scswitch -e -j IM-HASP-RS2
```

To configure the second instance of Instant Messaging Server:

1. Run the **configure** utility on the primary node.

```
# cd INSTALL-ROOTIM2/im
# ./configure
```

For more information about the **configure** utility, see *Instant Messaging Server Installation and Configuration Guide*.

2. When prompted for the Instant Messaging Server Runtime Files Directory, enter one of the following:

If you are using an **HAStoragePlus** for the runtime files, enter **/share-disk-dirIM2**.

3. When prompted for the Instant Messaging Server host name, enter the logical host.

For example, accept the logical host even if the **configure** utility cannot connect to the specified host. The logical host resource might be offline when you invoke the **configure** utility.

4. Do not start Instant Messaging Server after configuration or on system startup.

In an HA configuration, the Instant Messaging Server service requires the logical host to be online for Instant Messaging Server to work correctly.

5. Copy the Instant Messaging Server configuration file **iim.conf.xml** to the **iim.conf** file with the same permissions.

Note: Also copy the **iim.conf.xml** file to **iim.conf** after any future configuration changes as cluster uses the **iim.conf** file.

6. To use the **GatewayConnector** service in HA, update this service configuration with the virtual host name or IP address and port number as follows:

```
InstantMessaging_home/imconfutil --config config_file_path iim_
gwc.hostport=virtual host-name or ip:port
```

For example:

```
/opt/sun/comms/sbin/imconfutil --config /DATA1/default/config/iim.conf.xml iim_
gwc.hostport=192.10.12.11:33333
```

7. Create the Instant Messaging Server resource and enable the resource.

In this example, the resource group name is **IM_SVR_RS2**. Provide the logical host resource name, the **HAStoragePlus** resource name, and the port number. By default, Instant Messaging Server uses ports 5269, 5222, and 45222. If the first instance uses these port numbers, use different port numbers for the second instance.

```
/INSTALL-ROOTIM2/im/sbin/imconfutil --config /MS_ALTROOT/im/config/iim.conf.xml
set-prop iim_server.port=5270
/INSTALL-ROOTIM2/im/sbin/imconfutil --config /MS_ALTROOT/im/config/iim.conf.xml
set-prop iim_server.muxport=45223
/INSTALL-ROOTIM2/im/sbin/imconfutil --config /MS_ALTROOT/im/config/iim.conf.xml
set-prop iim_mux.listenport=5223
/INSTALL-ROOTIM2/im/sbin/imconfutil --config /MS_ALTROOT/im/config/iim.conf.xml
set-prop iim_mux.serverport=45223
scrgadm -a -j IM_SVR_RS2 -g IM-RG2
-t SUNW.iim -x Server_root=/INSTALL-ROOTIM2/im
-y Confdir_list=/share-disk-dirIM2/default/config
```

```
-y Resource_dependencies=IM-HASP-RS2,LOG-HOST-IM-RS2
```

8. Test the successful creation of the Instant messaging resource group by performing a failover.

```
scswitch -z -g IM-RG2 -h IM_NODE1
```

Note: You do not have to configure the second node as configuration is shared between all the nodes by soft links pointing to shared location.

Removing HA for Instant Messaging Server

To remove Instant Messaging Server from an HA environment, remove the Instant Messaging Server cluster agent **SUNWiimsc**.

When you remove the **SUNWiimsc** package as described in this procedure, any customization you made to the RTR file **SUNW.iim** is lost. If you want to restore them at a later time, you must create a backup copy of **SUNW.iim** before removing **SUNWiimsc**.

To remove HA for Instant Messaging Server:

1. Stop the Instant Messaging Server data service.

```
scswitch -F -g IM_RG
```

2. Disable all resources in the Instant Messaging Server resource group **IM_RG**.

```
scswitch -n -j IM_SVR_RS
scswitch -n -j LOG_HOST_RS
scswitch -n -j IM-HASP-RS
```

3. Remove the resources from the Instant Messaging Server resource group.

```
scrgadm -r -j IM_SVR_RS
scrgadm -r -j LOG_HOST_RS
scrgadm -r -j IM-HASP-RS
```

4. Remove the Instant Messaging Server resource group.

```
scrgadm -r -g IM_RG
```

5. Remove the Instant Messaging Server resource type.

```
scrgadm -r -t SUNW.iim
```

6. Remove the **SUNWiimsc** package by using the Sun Java Enterprise System installer or run the **pkgrm SUNWiimsc** command.

When you remove the package, any customization that you make to the RTR file is lost.

7. Remove any links that you have created during the HA configuration, if you are using a shared directory for configuration files and binaries.

```
rm /etc/opt/sun/comms/im
```

Configuring LDAP Failover

This chapter describes how to configure LDAP failover for Oracle Communications Instant Messaging Server on a multi-master replication (MMR) setup of LDAP servers.

Overview of Configuring LDAP Failover

LDAP failover in Instant Messaging Server enables you to configure the Instant Messaging server to have multiple LDAP servers as back-end storage. If one LDAP server becomes unavailable, the Instant Messaging server is able to fail over to another LDAP server.

LDAP failover works on a MMR setup of LDAP servers. All the LDAP servers in the settings are masters and have permission to read and write data. The Instant Messaging server uses only one server at a time but fails over to another LDAP server when the current server becomes unavailable. The other LDAP server is expected to be in sync with the current server as far as data is concerned.

Setting Up LDAP Failover

To set up an LDAP failover:

1. Set up the MMR with the LDAP Servers. All the LDAP servers should be master servers. That is, each server should have the permission to read and write data to all the LDAP servers.
2. Ensure that all the master servers in the setup are started and synchronized.
3. Use the **imconfutil** command to add the LDAP replicas, and LDAP server names and ports.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml add-ldap-replica  
id=ldap1 host=ldap1.example.com port=389  
imconfutil -c InstantMessaging_home/config/iim.conf.xml add-ldap-replica  
id=ldap2 host=ldap2.example.com port=489
```

4. Set the **iim_ldap.debugPool** property to **true**.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_  
ldap.debugPool=true
```

Note: Only the Instant Messaging server is replica aware. All the support tools that use the **iim.conf.xml** file are not replica aware. For support tools to start, the default LDAP server should be up and running.

Note: The default LDAP configuration in Instant Messaging Server is required, even when LDAP failover is configured. Example of default LDAP configuration:

```
iim_ldap.host=xyz:389  
iim_ldap.usergroupbinddn=cn=Directory Manager  
iim_ldap.usergroupbindcred=password
```

Managing Archiving in Instant Messaging Server

This chapter explains how to configure and manage email, file, message, and custom archiving for Oracle Communications Instant Messaging Server.

About Archiving

Instant message archiving can be done in the following ways:

- Email Archive. When using this method, chat and conference participants receive email containing the contents of the Instant Messaging Server sessions in which they participated. End users can use any email client to search and manage instant messages.
- File Archive allows you to archive the contents of a file that is transferred from one client to another.
- Message Archiving allows you to archive all the message data that passes through the server in any one-to-one or a group-chat conversation.
- Custom Archive. You can choose to use either the Instant Messaging Server archive providers, or create your own custom archive provider. Instant Messaging Server provides the APIs and SPIs that you use to write custom archive providers. For more information on Instant Messaging Server APIs, see "[Using the Web Presence API](#)" and "[Instant Messaging Server APIs](#)".

Regardless of which type of archive provider you choose to use, you must enable the archive provider by running the **imconfutil** command to configure the appropriate property.

You can configure Instant Messaging Server to use one or both archive methods at the same time. Additionally, you can enable the archive so that users can retrieve their messages.

Enabling and Disabling Archiving for Instant Messaging Server

Regardless of whether you choose to use email, a custom archive, or any combination of archives, you enable the archiving capability in Instant Messaging Server the same way as described in this section. Disabling archiving as described in this section disables all archives.

Enabling Instant Messaging Server Archiving

After you enable archiving for Instant Messaging Server, you must enable the archive provider for the type of archive you want to use as described in the following sections:

- [Enabling the Instant Messaging Server Email Archive](#)
- [Enabling a Custom Archive Provider](#)
- 1. Use the **imconfutil** command to set the **iim_server.msg_archive** property to **true**.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_server.msg_archive=true
```

- 2. Restart the server.

```
imadmin refresh server
```

Disabling Instant Messaging Server Archiving

This procedure disables all archiving for Instant Messaging Server. If you want to disable only email archiving or a custom archive you have configured, see one of the following sections:

- [Disabling the Instant Messaging Server Email Archive Provider](#)
- [Disabling a Custom Archive Provider](#)
- 1. Use the **imconfutil** command to set the **iim_server.msg_archive** configuration property to **false**.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_server.msg_archive=false
```

- 2. Restart the server.

```
imadmin refresh server
```

Archiving in Instant Messaging Server

This section describes the archiving mechanisms in Instant Messaging Server.

Managing Instant Messaging Server File Archive

The Archive Provider API allows you to archive the contents of a file that is transferred from one client to another by using an In-Band Bytestreams file transfer feature. The abstract class exposed by Instant Messaging Server to implement file archiving is **com.sun.im.provider.ByteStreamFilter**.

Implementing the Custom File Architecture Provider

Following is an example of implementing the custom provider:

```
package com.sun.im.provider;
/**
 * Custom file archival provider must extend this abstract
 * class
 */
public abstract class ByteStreamFilter
{
    /**
     * process a data block contained in a stream.
```



```

*
* This method needs to be overridden in order to perform
* archiving
*
* @param stream byte stream handle
* @param block block of bytes to be transferred.
*
*/
public void processData(ByteStream stream,
    ByteStreamBlock block)
{
    block.commit();
}
/**
 * called when a new byte stream is open
 *
 * @param from data originator address, uses xmpp address
 * @param to data recipient address, uses xmpp address format
 * @param stream byte stream handle
 */
public void openStream(String to, String from,
    ByteStream stream)
{
}
/**
 * called when a new byte stream is closed
 *
 * @param stream byte stream handle
 */
public void closeStream(ByteStream stream)
{
}
}

```

File Archiver Provider Example

The Instant Messaging Server enables you to write custom archive providers. The custom archive provider for file archiving stores the contents of the file that is transferred from one client to another to a local file. To write a custom archive provider, you must override methods in the abstract classes **ByteStreamFilter** and **ArchiveProvider**. **ByteStreamFilter** is used for archiving file transfers.

Following is an example of a custom archive provider:

```

package com.sun.im.provider;
public class FileArchiving extends ByteStreamFilter
{
    private FileWriter fstream;
    private BufferedWriter out;
    private StringBuffer buffer;
    public void processData(ByteStream stream,
        ByteStreamBlock block)
    {
        log.debug( {{FileArchiving}}:processData() called );
        String data = new String(block.getBytes());
        buffer += data;
    }
    public void openStream(String to, String from,
        ByteStream stream)
    {

```

```
log.debug( {{FileArchiving}}:openStream() called );
fstream = new FileWriter("/tmp/{{FileArchiving}}");
out = new BufferedWriter(fstream);
}
public void closeStream(ByteStream stream)
{
log.debug( {{FileArchiving}}:closeStream() called );
out.write(buffer.toString());
}
}
```

Compiling the Custom File Archival Provider Application

Compile your custom archive by including the **improvider.jar** file in your **classpath**.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.classpath=your-custom-provider
```

Enabling and Disabling the Instant Messaging Server File Archive Provider

This section describes how to enable and disable the Instant Messaging Server file archive provider.

Enabling File Archiving

To enable file archiving:

1. Enable file archiving.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.filter.enable=true
```

2. Enable your custom file archive provider.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.filters=fully-qualified-name-of-your-custom-provider-class
```

3. Restart the server.

```
imadmin refresh server
```

Disabling File Archiving

To disable file archiving:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.filter.enable=false
```

Managing Instant Messaging Server Message Archive

Message Archiving allows you to archive all the message data that passes through the server in any one-to-one or a group-chat conversation. The abstract class exposed by Instant Messaging Server to implement message archiving is **com.sun.im.provider.ArchiveProvider**.

Implementing the Custom Message Archival Provider

Following is an example of implementing the custom message archival provider:

```
/**
 * Custom message archival provider must extend this
 * abstract class
 */
package com.sun.im.provider;
public abstract class ArchiveProvider
{
    /**
     * invoked when a user signs on
     * @param uid identifier of the authenticated user
     */
    public void onLogin(String uid)
    {
    }
    /**
     * invoked when a user signs off
     * @param uid user identifier
     */
    public void onLogout(String uid)
    {
    }
    /**
     * invoked when a user creates a private conference
     * @param conferenceAddress address of the conference
     * @param uid unique identifier of the user who setup
     * the conference
     */
    public void onSetup(String conferenceAddress, String uid)
    {
    }
    /**
     * invoked when a user joins a conference
     * @param conferenceAddress address of the conference
     * @param uid unique identifier of the new participant
     */
    public void onJoin(String conferenceAddress, String uid)
    {
    }
    /**
     * invoked when a user leaves conference
     * @param conferenceAddress address of the conference
     * @param uid unique identifier of the leaving participant
     */
    public void onLeave(String conferenceAddress, String uid)
    {
    }
    /**
     * invoked when a private conference is terminated
     * @param conferenceAddress address of the conference
     * @param uid identifier of the user who closed the
     * conference
     */
    public void onClose(String conferenceAddress, String uid)
    {
    }
    /**
     * invoked when a user creates a private conference
     * @param conferenceAddress address of the conference
     * @param message invite message
     */
}
```

```
public void onInvite(String conferenceAddress,
com.sun.im.service.ReadOnlyMessage message)
{
}
/**
 * invoked when a message of type normal, headline or error is
 * received by the server. When a chat message is received in a
 * one-to-one or a group-chat conversation, onConferenceMessage
 * is used instead. Once archived, the message is visible only
 * to the originator and recipients of the message.
 * The originator and recipients addresses, message
 * identifier, message content, and other message attributes
 * can be obtained
 * using the methods in the com.sun.im.service.ReadOnlyMessage
 * interface.
 * @param message message
 */
public void onMessage(com.sun.im.service.ReadOnlyMessage message)
{
}
/**
 * invoked when a message of type normal, headline or error is
 * received by the server. When a chat message is received in a
 * one-to-one or a group-chat conversation, onConferenceMessage
 * is used instead. Once archived, the message is visible only
 * to the originator and recipients of the message.
 * The originator and recipients addresses, message
 * identifier, message content, and other message attributes
 * can be obtained
 * using the methods in the com.sun.im.service.ReadOnlyMessage
 * interface.
 * @param message message
 */
public void onMessage(com.sun.im.service.ReadOnlyMessage message)
{
}
/**
 * invoked when a message is received by the server in any
 * one-to-one or a group-chat conversation.
 * @param conferenceAddress address of the conference
 * @param message message
 * The originator address, message identifier,
 * message content, and other message attributes can be
 * obtained using the methods in the Message interface.
 * @see com.sun.im.service.Message
 */
public void onConferenceMessage(String conferenceAddress,
com.sun.im.service.ReadOnlyMessage message)
{
}
/**
 * Add code store to marker messages
 */
@Override
public void onMarkerMessage(ReadOnlyMarkerMessage mm)
{
}
/**
 * open the archive
 * @exception Exception failure to open and initialize the
```

```
* archive.
*/
public void open() throws Exception
{
}
/**
 * close the archive and dispose off all held resources
 */
public void close()
{
}
}
```

Message Archive Provider Example

The custom message archive provider example for message archiving stores the messages exchanged in any one-to-one or group-chat conversation in a local file. It also logs invite messages and join and leave events of a group-chat. **ArchiveProvider** is used to archive one-to-one chat and group chat messages. **ArchiveProvider** for both files and messages are enabled using the **imconfutil** command.

Following is an example of an custom message archive provider:

```
package com.sun.im.provider;
public class MessageArchiving extends ArchiveProvider
{
    private FileWriter fstream;
    private BufferedWriter out;
    public void onLogin(String uid)
    {
        log.debug( MessageArchiving:onLogin() called );
    }
    public void onLogout(String uid)
    {
        log.debug( MessageArchiving:onLogout() called );
    }
    public void onSetup(String conferenceAddress,
        String uid)
    {
        log.debug( MessageArchiving:onSetup() called );
        out.write( Conference has been created );
    }
    public void onJoin(String conferenceAddress, String uid)
    {
        log.debug( MessageArchiving:onJoin() called );
        out.write(uid +   has joined the conference );
    }
    public void onLeave(String conferenceAddress, String uid)
    {
        log.debug( MessageArchiving:onLeave() called );
        out.write(uid +   has left the conference );
    }
    public void onClose(String conferenceAddress,
        String uid)
    {
        log.debug( MessageArchiving:onClose() called );
        out.write( Conference has been closed );
    }
    public void onInvite(String conferenceAddress,
        com.sun.im.service.ReadOnlyMessage message)
```

```
{
log.debug( MessageArchiving:onInvite() called );
out.write(message.getOriginator() + has invited + message.getRecipients() +
to the conference );
}
public void onMessage(com.sun.im.service.ReadOnlyMessage message)
{
log.debug( MessageArchiving:onMessage() called );
}
public void onMessage(java.util.List accessList,
com.sun.im.service.ReadOnlyMessage message)
{
log.debug( MessageArchiving:onMessage() called );
}
public void onConferenceMessage(String conferenceAddress,
com.sun.im.service.ReadOnlyMessage message)
{
log.debug( MessageArchiving:onConferenceMessage() called );
out.write(message.getOriginator()+ sent a message, +message.getContent()+ to
+message.getRecipients());
}
public void onMarkerMessage(ReadOnlyMarkerMessage mm)
{
log.debug( MessageArchiving:onMarkerMessage() called );
}
public void open() throws Exception
{
log.debug( MessageArchiving:open() );
fstream = new FileWriter("/tmp/MessageArchiving");
out = new BufferedWriter(fstream);
}
public void close()
{
log.debug( MessageArchiving:close() );
}
}
```

Compiling the Custom Message Archival Provider Application

Compile your custom archive using the following jar file in the **classpath**:
imservice.jar.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.classpath=your-custom-provider
```

Enabling and Disabling the Instant Messaging Server Message Archive Provider

This section describes how to enable and disable the Instant Messaging Server message archive provider.

To enable message archiving:

1. Run the following **imconfutil** command.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.msg_archive=true
```

2. Enable your custom message archive provider.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.msg_archive.provider=fully-qualified-name-of-your-custom-provider-class
```

3. Restart the server.

```
imadmin refresh server
```

To Disable Message Archiving

To disable message archiving, run the following **imconfutil** command.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_server.msg_archive=false
```

Managing Instant Messaging Server Email Archive

You can use Instant Messaging Server to archive chat and conference, content, and email that content to end-users and administrators. You can use any email client to search and manage the archived content.

Instant Messaging Server caches archived records until they are emailed. If you enable email archiving, the memory requirements for the server increase. See ["Improving Instant Messaging Server Performance"](#) for information on performance tuning.

Enabling and Disabling the Instant Messaging Server Email Archive Provider

You enable or disable the email archive provider by modifying a the appropriate configuration property.

Enabling the Instant Messaging Server Email Archive

Ensure that you have enabled archiving for Instant Messaging Server. See ["Enabling Instant Messaging Server Archiving"](#) for more information.

1. Use the **imconfutil** command to set the **iim_server.msg_archive.provider** property.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_server.msg_archive.provider=com.ipplanet.im.server.EmailIMArchive
```

The **iim_server.msg_archive.provider** property contains a comma-separated list of archive providers.

2. Restart the server.

```
imadmin refresh
```

Disabling the Instant Messaging Server Email Archive Provider

1. Use the **imconfutil** command to remove the **iim_server.msg_archive.provider** property.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml del-prop iim_server.msg_archive.provider
```

2. Restart the server.

```
imadmin refresh
```

Configuring Email Archive Settings

You can configure which administrators receive email containing archived instant messages. You can configure a separate list of administrators to receive conference or

chat sessions. You can also configure Instant Messaging Server to use the extended RFC 822 header. Doing so enables mail clients to filter messages based on the header content.

Note: If you run **configure** after modifying these properties for the email archive, any values you input are overwritten.

Table 10–1 describes the configuration properties that you use to define which administrators receive email archives, as well as whether to use the extended RFC 822 header, and the content of that header.

Table 10–1 Email Archive Configuration Properties

Property	Default Value	Description
<code>iim_arch.admin.email</code>	Empty String	Comma-separated list of administrator email addresses.
<code>emailiim_arch.alert.admin.</code>	None	Comma-separated list of administrator email addresses to which all archived alert messages are sent. This property overrides <code>iim_arch.admin.email</code> for alert messages.
<code>iim_arch.chat.admin.email</code>	None	Comma-separated list of administrator email addresses to which all archived chat messages are sent. This property overrides <code>iim_arch.admin.email</code> for chat messages.
<code>iim_arch.conference.admin.email</code>	None	Comma-separated list of administrator email addresses to which all archived conference messages are sent. This property overrides <code>iim_arch.admin.email</code> for conference messages.
<code>iim_arch.poll.admin.email</code>	None	Comma-separated list of administrator email addresses to which all archived poll messages are sent. This property overrides <code>iim_arch.admin.email</code> for poll messages.
<code>iim_arch.news.admin.email</code>	None	Comma-separated list of administrator email addresses to which all archived news messages are sent. This property overrides <code>iim_arch.admin.email</code> for news messages.
<code>iim_arch.email.archiveheader.name</code>	None	Name of the extended RFC 822 header.
<code>iim_arch.email.archiveheader.value</code>	all	Value corresponding to the header name for <code>iim_arch.email.archiveheader.name</code> .

Configuring Administrator Recipients and the RFC 822 Header Format

To configure administrator recipients:

1. Run the **imconfutil** command to add the properties in Table 10–1 and appropriate values to the configuration.
2. Restart the server.

```
imadmin refresh
```

Email Header Format

This section describes the RFC 822 header content for email messages containing various types of archived Instant Messaging Server content.

RFC 822 Email Archive Header Fields for One to One Chat

From: Chat session initiator.

To: Receiver and any administrators configured in `iim.conf.xml`.

See Table 18–1 for more information.

Subject: First useful message over 50 characters in length.
Date: Creation date of the email message by the archive provider.
Reply-to: Not used.
Message-ID Generated by the email archive provider based on the message thread.

RFC 822 Email Archive Header Fields for Private Conferences

From: Chat session initiator.
To: Other participants and any administrators configured in iim.conf.xml. See Table 18-1 for more information.
Cc: Chat session initiator.
Subject: If a subject is set for the conference, the conference subject is used. If no subject is set, first useful message over 50 characters in length is used.
Date: Creation date of the email message by the archive provider.
Reply-to: Not used.
X-XMPP-Message-ID Generated by the email archive provider based on the conference ID.

RFC 822 Email Archive Header Fields for Public Conferences

From: Room owner in archive data.
To: Associated mailing list, users with explicit access to the conference room, and any administrators configured in iim.conf.xml. See Table 18-1 for more information.
Cc: Not used
Subject: [Conference name] subject.
Date: Creation date of the email message by the archive provider.
Reply-to: Not used.
X-XMPP-Message-ID Generated by the email archive provider based on the conference ID.

RFC 822 Email Archive Header Fields for Poll Questions with Replies

From: Poll sender.
To: Poll sender and any administrators configured in iim.conf.xml. See Table 18-1 for more information.
Cc: Not used.
Subject: Poll question.
Date: Creation date of the email message by the archive provider.
Reply-to: Not used.
X-XMPP-Message-ID Generated by the email archive provider.

RFC 822 Email Archive Header Fields for Poll Replies Only

From: Poll sender.
To: Poll recipients and any administrators configured in iim.conf.xml. See Table 18-1 for more information.
Cc: Poll sender.
Subject: Poll question.

Date: Creation date of the email message by the archive provider.
Reply-to: Not used.
X-XMPP-Message-ID Generated by the email archive provider.

RFC 822 Email Archive Header Fields for Alerts

From: Alert sender.
To: Alert recipient and any administrators configured

in `iim.conf.xml`. See Table 18-1 for more information.
Cc: Not used.
Subject: Alert subject.
Date: Creation date of the email message by the archive provider.
Reply-to: Not used.
X-XMPP-Message-ID Generated by the email archive provider.

RFC 822 Email Archive Header Fields for New Channel Posts

From: News channel post sender.
To: Mailing list associated with the news channel
and any administrators configured in `iim.conf.xml`.
See Table 18-1 for more information.
Cc: Not used.
Subject: News channel post subject.
Date: Creation date of the email message by the archive provider.
Reply-to: Not used.
X-XMPP-Message-ID Generated by the email archive provider.

Enabling and Disabling the Instant Messaging Server Custom Archive Provider

In addition to the email archive, you can choose to use a custom archive provider.

Enabling a Custom Archive Provider

Ensure that you have enabled archiving for Instant Messaging Server. See ["Enabling Instant Messaging Server Archiving"](#) for more information.

To enable a custom archive provider:

1. Use the **imconfutil** command to add the type of archive provider you want to enable.

For example, for a custom archive provider:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.msg_archive.provider=provider-name
```

The **iim_server.msg_archive.provider** property contains a comma-separated list of archive providers. The following example enables the email provider.

```
iim_server.msg_archive.provider=com.iplanet.im.server.EmailIMArchive
```

2. Restart the server.

```
imadmin refresh
```

Disabling a Custom Archive Provider

To disable a custom archive provider:

1. Use the **imconfutil** command to delete only the value for the custom archive provider from the **iim_server.msg_archive.provider** property.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml del-prop iim_
server.msg_archive.provider=provider-name
```

2. Restart the server.

```
imadmin refresh
```

About Archived Messages Retrieval

Instant Messaging Server supports the retrieval of archived messages (one-to-one chats), even those which have been sent and received from multiple end points. Users can then retrieve these archived messages from the archival store upon login from any end point. Archiving is managed by using the Instant Messaging Server Archive Provider API. Currently, retrieval of conference messages and group messages is not yet available.

Archived Message Management Example

The custom message archive management provider is required for retrieving stored messages from the message archive. This provider extends the **ArchiveProvider** class and provides two additional methods that Instant Message Server invokes when it queries the message archive. In addition, this provider is required for automatic history synchronization between multiple clients.

Enabling Retrieval of Archived Messages

To enable retrieval of archived messages:

1. Set the **iim_server.msg_archive** property to **true**:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_server.msg_archive=true
```

2. Set the **iim_server.msg_archive.provider** property to the class name of your Archive Provider implementation:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_server.msg_archive.provider="com.example.archiving.MessageArchivingSample"
```

See "[Message Archive Retrieval Provider Example](#)" for information about writing a sample archiving implementation.

3. Set the **iim_server.msg_archive.maxstanzas** property to the maximum number of message stanzas to be retrieved from the archive store. (A stanza is a message packet which holds the message information.)

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_server.msg_archive.maxstanzas=10
```

4. Set the **iim_server.classpath** property to the location of the JAR file that contains the implementation of the provider:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_server.classpath=path_to_sample/MessageArchivingSample.jar
```

5. Restart Instant Messaging Server.

Message Archive Retrieval Provider Example

The following sample provider queries the message store.

```
public List<Message> getArchivedMessages(String user, SearchCriteria sc, int count) {
    // Add code to fetch messages for given criteria
    // Refer next method to create list of messages
}

public List<Message> getArchivedMessages(String user, int count) {
    // Add code to fetch messages
}
```

```
List<Message> list = new ArrayList<Message>;

MessageFactory fac = new MessageFactory()
Message message = fac.createMessage("aeb213", "juliet@capulet.lit/balcony",
"romeo@montague.lit/orchard", "Call me but love, and I'll be new baptized;
Henceforth I never will be Romeo." );
message.setHeader(MessageHeaders.ARCHIVE_UID_HEADER, "28482-98726-73623");
message.setHeader(MessageHeaders.FORWARDED_TIMESTAMP_HEADER,
"2010-07-10T23:08:25Z"); // stamp field of result message
list.add(message);
return list;
}
}
```

Each message that is part of the list contains an archive UID and a timestamp. The archive UID uniquely identifies the message in the archive store and is used for Result Set Management. The timestamp is in ISO 8061 format, which is the time at which the message was archived into the store. Both UID and timestamp are forwarded to the client.

The **MessageFactory** class enables message creation. This class is part of the **imservice.jar** file, which facilitates creation of chat messages and marker messages. You can add archive UID and timestamps of a message to the newly created message by using the **setHeader(String name, String value)** method of the 'Message' interface. The header constant to be used to set archive UID is **MessageHeaders.ARCHIVE_UID_HEADER**. The header constant to be used to set the timestamp is **MessageHeaders.FORWARDED_TIMESTAMP_HEADER**. You can use **setHeader** to set other message metadata such as thread information and subject.

Using Chat Markers

Chat markers are the visual indications between all the sender's and recipient's resources, indicating when a message has been delivered to any of the recipient's resources, and optionally when it has been displayed (read). In this way, chat markers help to consistently maintain the state of a chat across all end points. These marker messages are also stored to the archive store if message archiving is enabled. Additionally, if one or more of the resources is not connected, it can fetch chat markers from the message archive upon reconnecting.

The chat marker is in itself a kind of message that indicates the status of a message after it has been sent to a client. A chat marker status can be:

- Received - Indicates that a user received a message.
- Acknowledged - Indicates that the user who received the message acknowledged it.
- Displayed - Indicates that the message was displayed to the intended user.

The following code shows how chat markers can be applied to the storage and retrieval of one-to-one messages. Currently, they cannot be applied to group chat or conference messages.

```
public void onMarkerMessage(ReadOnlyMarkerMessage mm) {
    /**
     * Add code to store marker messages
     */
}
```

The **onMarkerMessage** method enables chat markers to be included in archived messages.

If the chat markers are stored in the archive store, create a marker message using the **MessageFactory** class then append the marker to message list. For example:

```
public List<Message> getArchivedMessages(String user, int count) {

    List<Message> list = new ArrayList<Message>;

    // create normal messages 1..n

    MarkerMessage marker = fac.createMarkerMessage("marker-1");
    marker.setMarker("aeb213", MarkerStatus.ACKNOWLEDGED);
    marker.setOriginator("romeo@montague.lit/orchard");
    marker.addRecipient("juliet@capulet.lit/balcony");
    message.setHeader(MessageHeaders.THREAD_HEADER, "thread1");
    list.add(marker);
    return list;
}
```

Managing Message Conversion in Instant Messaging

This chapter explains how to configure and manage message conversion for Oracle Communications Instant Messaging Server.

About Message Conversion

If enabled, a Message Converter is invoked for every message or each message part going through the server. The Message Converter may leave the message part intact or modify or remove the message part. Message Conversion can be used to manipulate the message packets that pass through the server. Message packets can either be from a one-to-one chat or from a group-chat conversation.

Managing Message Conversion in the Instant Messaging Server

Message Conversion can be used to manipulate the message packets that pass through the server. Message packets can either be from a one-to-one chat or from a group-chat conversation. The abstract class `com.sun.im.provider.MessageConverter` needs to be implemented to write a custom provider for Message Conversion.

Implementing the Custom Message Conversion Provider

To implement the conversion provider, you must extend the `MessageConverter` class and override the corresponding methods.

Following is an example of implementing a custom message conversion provider:

```
package com.sun.im.provider;
/**
 * Custom provider for Message Conversion must extend this abstract class.
 */
public abstract class MessageConverter
{
    /**
     * Convert a message part.
     * This method may make modification to the content, content-
     * type and content-name of the provided MessagePart object.
     *
     * @param part incoming message part to convert.
     * If the contents of the part once modified are null, the part is
     * removed.
     * @deprecated instead use com.sun.im.service.Message
     * @exception Exception the converter may throw an Exception.
     * If so the exception is logged in the server log file and the message
```

```
* is not relayed to any recipients.
* The sender receives a negative delivery status.
*/
public void convert(com.sun.im.service.MessagePart part) throws Exception
{
    return;
}
/**
 * Convert a message part.
 * This method may make modification to the content, content-
 * type and content-name of the provided MessagePart object.
 * It needs to be overwritten by actual message converters.
 * The default behaviour of this method is to call
 * convert(com.sun.im.service.MessagePart)
 * so all the extensions to MessageConverter written prior
 * to version 7.0 will still work with later versions.
 * @param message incoming message to convert. If the contents
 * of the message once modified are null, the message is
 * removed. Implementing the Custom Message Conversion Provider
 * If so the exception is logged in the server log file and
 * the message is not relayed to any recipients.
 */
public void convert(com.sun.im.service.Message message) throws Exception
{
    com.sun.im.service.MessagePart parts[] = message.getParts();
    convert(parts[0]);
}
}
```

Message Converter Provider Example

Following is an example of a custom message converter provider:

```
public class MessageConverterSample extends MessageConverter
{
    private FileWriter fstream;
    private BufferedWriter out;
    static {
        try {
            fstream = new FileWriter("/tmp/MessageConverter");
            out = new BufferedWriter(fstream);
        } catch (IOException ex) {
            Logger.getLogger(MessageConverterSample.class.getName())
                .log(Level.SEVERE, null, ex);
        }
    }
    public void convert(com.sun.im.service.Message message) throws Exception
    {
        Log.debug( MessageConverter:convert() called );
        String convertedMessage = message.getContent() + " \
        \nDISCLAIMER : Messages are archived for security reasons";
        message.setContent(convertedMessage);
        out.write( Converted message is + message.getContent());
    }
}
```

Compiling the Custom Message Converter Provider

Compile your custom message converter provider using the following JAR file in the classpath: **imservice.jar**.

Use the **imconfutil** command to include the JAR file in your **classpath**.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.classpath=your-custom-provider
```

Enabling and Disabling the Instant Messaging Message Converter Provider

This section describes how to enable and disable the Instant Messaging Server message converter provider.

To enable message conversion and the message converter provider:

1. Enable message conversion.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.conversion=true
```

2. Enable your custom message converter provider.

```
imconfutil -c InstantMessaging_home/iim.conf.xml set-prop iim_
server.conversion.provider=fully-qualified-name-of-your-custom-provider
```

3. Restart the server.

```
imadmin refresh server
```

To disable message conversion, use the following command:

```
imconfutil -c InstantMessaging_home/iim.conf.xml set-prop iim_
server.filter.enable=false
```

Monitoring Instant Messaging Server and Multiplexor

This chapter describes how to collect data and monitor Oracle Communications Instant Messaging Server and Instant Messaging multiplexor activity.

Overview of Monitoring the Instant Messaging Server

You can collect data and monitor Instant Messaging Server activity. Examples of the types of data you can collect are:

- Total number of users currently online.
- Average number of message transfers, per second, in a fixed period of time.
- Amount of memory consumed by the server process (XMPPd).
- Transaction times for login, roster fetch, and sending an instant message.
- Number of packets exchanged with each federated domain.

You can use the collected data for purposes such as:

- Monitoring the health of the server and identifying quality of service issues.
- Resource utilization and capacity planning.
- Gathering per-domain usage data for billing purposes.

See "[Available Server Metrics](#)" for more information on the statistics that can be collected.

To monitor the Instant Messaging data collected you can use either of the following:

- Oracle Enterprise Manager (see "[Installing and Configuring the Oracle Enterprise Manager Plug-in](#)").
- A JMX client, such as JConsole. See "[Available Server Metrics](#)" for information on the JMX metrics available for monitoring. For information on accessing the collected metrics, see the Jconsole documentation at:

<http://docs.oracle.com/javase/7/docs/technotes/guides/management/jconsole.html>

Configuring Instant Messaging Server Monitoring

To configure data collection and monitoring, you must enable server monitoring and configure JMX-based server monitoring, the JVM, and the JAAS. For more information on JMX and JAAS settings and configuration files, see the JMX documentation at:

<http://docs.oracle.com/javase/7/docs/technotes/guides/management/agent.html>

Steps for Configuring Data Collection and Monitoring

All configuration in the following steps is carried out through the **imconfutil** command.

1. Log in to the server as **root**.
2. Enable monitoring by setting the **iim_server.monitor.enabled** property to **true**:

```
imconfutil set-prop iim_server.monitor.enable=true -c InstantMessaging_
home/config/iim.conf.xml
```

3. Set the user name and password for the server to use in JAAS authentication:

```
imconfutil set-prop iim_server.admin.user=UserID -c InstantMessaging_
home/config/iim.conf.xml
imconfutil set-prop iim_server.admin.password=Password -c InstantMessaging_
home/config/iim.conf.xml
```

You must also assign the user name you enter **readwrite** permission in the JMX access-control file.

4. Configure JVM and JAAS properties for monitoring.

The Instant Messaging Server stores all JVM and JAAS properties in a single configuration property that is passed to the server process. [Table 12–1](#) lists the properties you must set.

Table 12–1 Property Settings for Configuring JMX and JAAS for Server Monitoring

Property	Description
-Dcom.sun.management.jmxremote	Enables remote monitoring.
-Dcom.sun.management.jmxremote.port=port_number	Sets the JMX remote access port for the client connection.
-Dcom.sun.management.jmxremote.local.only=false	Allows remote JMX access.
-Dcom.sun.management.jmxremote.ssl=true	true enables SSL for the JMX connection, false disables it.
-Dcom.sun.management.jmxremote.authenticate=true	Enables JAAS authentication.
-Djava.security.auth.login.config=InstantMessaging_home/config/jaas_login_config_file	Specifies the location of the JAAS login configuration file to use for authentication. For more information, see the JAAS documentation at: http://docs.oracle.com/javase/7/docs/technotes/guides/security/jgss/tutorials/LoginConfigFile.html
-Dcom.sun.management.jmxremote.login.config=JAAS_Login	Specifies the name of JAAS login entry in the JAAS login configuration file. The name of the entry must match the name you enter for -Dcom.sun.management.jmxremote.login.config with the imconfutil command.
-Dcom.sun.management.jmxremote.access.file=InstantMessaging_home/config/jmxaccess	Specifies the absolute path to the JMX access file. You can create a jmxaccess file based on the JMX-access template file located at: <i>JDK_InstallDir\jre\lib\management\jmxremote.access</i> . In configuring the jmxaccess file, the name of the user specified in controlRole must be same as the user name you entered for the server to use in JAAS authentication.

Table 12–1 (Cont.) Property Settings for Configuring JMX and JAAS for Server Monitoring

Property	Description
-Dcom.sun.management.jmxremote.ssl=true	true enables SSL for the JMX connection, false disables it. If you enable SSL, supply the keystore and password as well.
-Djavax.net.ssl.keyStore	Specifies the location of the keystore.
-Djavax.net.ssl.keyStorePassword	Specifies the location of the keystore password.
-Dcom.sun.management.jmxremote.ssl.need.client.auth=false	Disables mutual SSL authentication.

You must set all of the properties in a single **imconfutil** command. The **imconfutil** command in the following example sets the JVM and JAAS properties required for monitoring.

```
imconfutil set-prop iim_
server.jvm.options="-Dcom.sun.management.jmxremote.access.file=/opt/sun/comms/i
m/config/jmxaccess -Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=9010
-Dcom.sun.management.jmxremote.local.only=false
-Dcom.sun.management.jmxremote.authenticate=true
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.login.config=Login1
-Djava.security.auth.login.config=/opt/sun/comms/im/config/jaasconfig" -c
InstantMessaging_home/config/iim.conf.xml
```

5. Add an entry to the JAAS login configuration file that contains the **com.oracle.im.stat.AdminLoginModule** implementation of **LoginModule**. For information, see the topic on configuring JAAS login entries at:

<http://docs.oracle.com/javase/7/docs/technotes/guides/security/jaas/JAASRefGuide.html#AppendixB>

The name of the entry must match the name you enter for

-Dcom.sun.management.jmxremote.login.config with the **imconfutil** command.

6. Set the interval, in seconds, at which you want monitored data to be refreshed. The default interval is 30 seconds. To configure an alternative interval:

- Set the **iim_server.monitor.refreshtimeout** property as in the following example:

```
imconfutil set-prop iim_server.monitor.refreshtimeout=60 -c
InstantMessaging_home/config/iim.conf.xml
```

- If you are using a JMX client, you can use the **refreshNow** operation in the JMX Statistics MBean for an immediate refresh.

Installing and Configuring the Oracle Enterprise Manager Plug-in

The Oracle Enterprise Manager Plug-in for Instant Messaging Server extends the Enterprise Manager to provide access to the following items for monitoring Instant Messaging Server:

- Usage and performance metrics by server and domain
- Alerts and notifications based on thresholds that you set on monitored characteristics.

Product Version Requirements

The Enterprise Manager Plug-in for Oracle Communications Instant Messaging Server supports:

- Enterprise Manager Cloud Control 12c Release 1 (12.1.0.3.0) or higher
- Oracle Instant Messaging Server 9 Update 1 or higher

Installation Prerequisites

You must install the following products before you can install the plug-in:

- Enterprise Manager Cloud Control 12c Release 1 (12.1.0.3.0) or higher. For more information, see *Oracle Enterprise Manager Cloud Control Basic Installation Guide*.
- Oracle Instant Messaging Server 9 Update 1 or higher

Downloading the Enterprise Manager Plug-in

Download the plug-in from the Enterprise Manager Store as follows:

1. Log in to the Enterprise Manager Cloud Control administration console as a privileged user.
2. From the **Setup** menu, select **Extensibility**, then **Self Update**.
3. Select the **Plug-in** row and click **Open**.
4. Scroll down and select the **Oracle Communications Instant Messaging Server** plug-in.
5. Click **Download**.

The plug-in becomes deployable once the download completes. See "[Deploying the Enterprise Manager Cloud Control Plug-in](#)" for information on deploying the plug-in.

Configuring Instant Messaging Server Targets

You monitor Instant Messaging Server hosts and domains setup as managed targets in Enterprise Manager. Enterprise Manager also monitors other component non-host targets installed in your environment. Non-host targets consist of applications and their components and infrastructure, for example, Oracle Enterprise databases.

An Oracle Management Agent runs on each host with one or more targets. The Enterprise Manager Management Server communicates with the management agent performing monitoring of host and non-host targets. You must install the management agent on any host you plan on using with Instant Messaging Server. For more information on managed targets, see the topic on discovering and monitoring targets in *Oracle Enterprise Manager Cloud Control Administrator's Guide*.

Adding Instant Messaging Server Host Targets and Installing the Management Agent

Add each Instant Messaging Server host you want to monitor to Enterprise Manager as a managed target and install a management agent manually on each host.

To add an individual host and install a management agent on it:

1. Log in to the Enterprise Manager administration console.
2. Expand the **Setup** menu, then select **Add Target**, then **Add Targets Manually**.
3. Select **Add Host Targets**.

4. Click **Add Host**.
5. Click **Add** in the **Add Host Targets: Host and Platform** wizard.
6. Enter the new target's host name in **Host**, then select the correct operating system platform from the **Platform** menu.
7. Click **Next**.
8. Enter an **Installation Base Directory** on the new target.
9. Enter an **Instance Directory** on the new target.
10. Select a **Named Credential for Management Agent** installation on the new target. See ["Setting Up Preferred Credentials"](#) for more information on setting up host credentials.
11. Confirm the **Privileged Delegation Setting** and **Port**, as well as any **Optional Details** needed in your installation.
12. Click **Next**.
13. Confirm the **Host Information** and click **Deploy Agent**.
14. Confirm that the **Management Agent** is properly installed and the new target is now visible in the administration console.

For detailed information on installing the Management Agent, see *Oracle Enterprise Manager Cloud Control Basic Installation Guide*.

Setting Up Preferred Credentials

Enterprise Manager Cloud Control uses preferred credentials for authentication between the management server and managed agents. You set either default credentials for particular target types or target-specific credentials that are stored in the Enterprise Manager Cloud Control repository.

For more information on setting up preferred credentials, see the discussion on preferred credentials in *Oracle Enterprise Manager Cloud Control Administrator's Guide*.

Access the Enterprise Manager Cloud Control Security credential store and configure preferred credentials using the following procedure:

1. Log in to the Enterprise Manager Cloud Control administration console.
2. Click **Setup**, then **Security**, then **Preferred Credentials**.
3. Click **Manage Preferred Credentials**.
4. To set default preferred credentials, click **Set** under Default Preferred Credentials. To use target preferred credentials, click **Set** under Target Preferred Credentials.

You can also set up target preferred credentials when adding a new managed host.

Deploying the Enterprise Manager Cloud Control Plug-in

You must deploy the plug-in on both the management server host and on all Instant Messaging Server hosts running a management agent.

- [Deploying the Enterprise Manager Cloud Control Plug-in on the Management Server](#)
- [Deploying the Enterprise Manager Cloud Control Plug-in on Host Targets](#)

Deploying the Enterprise Manager Cloud Control Plug-in on the Management Server

To deploy the plug-in on the management server:

1. Set up preferred credentials for the target host. See ["Setting Up Preferred Credentials"](#) for more information.
2. From the **Setup** menu, select **Extensibility** then select **Plug-ins**.
3. Select **Oracle Instant Messaging Server** from the **Applications** folder.
4. From the **Deploy On** menu, select **Management Servers**.
5. In the **Deploy Plug-in on Management Servers** dialog, enter the password for the **Sys** user and click **Continue**.
6. Complete the remaining steps in the dialog box.
7. Click **Deploy**.
8. Monitor the status to ensure successful deployment.

Deploying the Enterprise Manager Cloud Control Plug-in on Host Targets

Deploy the Enterprise Manager Cloud Control Plug-in for Oracle Communications Instant Messaging Server to all host management agents you manage with Enterprise Manager Cloud Control using the following process:

1. Log in to the Enterprise Manager Cloud Control administration console.
2. Expand the **Setup** menu, then select **Extensibility**, then **Plug-Ins**.
3. Expand **Applications**.
4. Right-click **Oracle Instant Messaging Server**.
5. Select **Deploy On**, then **Management Agent**.
6. In the **Deploy Plug-in on Management Agent** screen, confirm the **Target Type** and click **Continue**.
7. Click **Add** under Select Management Agent in the pop-up window.
8. Select the target(s) on which to deploy the plug-in.
9. Click **Select**.
10. Click **Continue**.
11. Confirm there are no errors in the pre-requisite check.
12. Click **Next**.
13. Click **Deploy**.
14. Confirm that the **Enterprise Manager Cloud Control Plug-in for Oracle Communications Instant Messaging Server** deploys successfully.
15. Repeat these steps for all Instant Messaging Server hosts.

For additional information on deploying plug-ins, refer to the topic on Plug-in Manager in *Oracle Enterprise Manager Cloud Control Administrator's Guide*.

Using the Enterprise Manager to Monitor Instant Messaging Server

Instant Messaging Server metrics consist of JMX-collected data that provides information on the usage and status of your implementation. See ["Available Server"](#)

[Metrics](#)" for a list of the metrics available. The metrics have default thresholds for generating alerts and sending notifications. See ["Setting Thresholds on Monitored Metrics"](#) to change the thresholds. By default, no corrective actions are initiated when a threshold is crossed. To configure corrective actions, see ["Adding Corrective Actions"](#).

Viewing Metrics

You can monitor metrics from managed target instances of Instant Messaging Server through the Enterprise Manager Cloud Control administration console:

1. Log in to the Enterprise Manager Cloud Control administration console as a privileged user.
2. Click **Targets**, then **All Targets**.
3. In the list of targets, click the **Instant Messaging Server** in the Instant Messaging Server column you want to monitor.

Instant Messaging Server targets have an **Oracle Instant Messaging Server - Core Server** target type. Enterprise Manager Cloud Control displays the target's overview page.

4. Expand the **Oracle Instant Messaging Server - Core Server** menu under the *target_name*.
5. Click **Monitoring**, then **All Metrics**.
6. In the left-hand tree control, expand the metric category and select the metric you want to view.

Enabling and Using the Beacon Service

A *beacon* is a component within the Management Agent that executes tests at regular intervals. A service is considered available if the test executes successfully on at least one key beacon.

To enable the beacon service:

1. Log in to Oracle Enterprise Manager.
2. From the **Targets** menu, select **Service** and then click create **Generic Service**.
3. Enter the required information to define the service availability based on the execution of the service test by the key beacons.
4. Select the **Service test** and enter the information for the service test, test type as XMPP, and test parameters.
5. Add the EM Management Beacon for executing the tests.
6. Add the Performance Metric based on Service Test and set threshold values to get incidents.
7. Review the information and click **Finish**.

To use the beacon service:

1. Go the Home page of created service.
2. Select **Administration**.
3. Select **service tests and beacons**.

Available Server Metrics

Table 12–2 lists the available metrics and gives default thresholds for generating alerts and sending notifications. It also gives the interval at which each metric is refreshed. The table lists metrics based on their organization in the Oracle Enterprise Manager.

Table 12–2 Available Server Metrics

Enterprise Manager Metric	JMX Parameter	Description
Active Users (in domain)	C2SCountPerDomain	<p>The number of active Instant Messaging Server users in the domain.</p> <ul style="list-style-type: none"> ■ Warning Threshold: Greater than 80000 users ■ Critical Threshold: Greater than 100,000 users ■ Collection Schedule: Every 10 Minutes
Active Users (logged in)	C2STypeDistroMap	<p>The number of active Instant Messaging Server logins. Separate values are given for MUX, XMPP, and HTTP, as follows:</p> <ul style="list-style-type: none"> ■ MUX: The number of clients connected through the Multiplexor. ■ XMPPD: The number of clients connected directly to the server (this is not a recommended configuration). ■ HTTP: The number of clients connected through HTTPBIND. <p>The thresholds and collection schedule for this metric are:</p> <ul style="list-style-type: none"> ■ Warning Threshold: Greater than 80000 users ■ Critical Threshold: Greater than 100,000 users ■ Collection Schedule: Every 10 Minutes
Server Overall Status (up/down)	None	<p>The status (up or down) of the Instant Messaging Server and of the Instant Messaging Multiplexor and Watchdog components, when they are running on the same machine as the Server:</p> <p>0: The Instant Messaging Server, Multiplexor, and Watchdog are up.</p> <p>1: The Server is down.</p> <p>2: The Multiplexor is down.</p> <p>3: The Watchdog is down.</p> <p>6: There was an error while trying to get the status of the services.</p> <p>If a component is not on the same machine as the Instant Messaging Server, it is not included in the status. For example, if the Multiplexor is running on a separate machine from the Instant Messaging Server, the status can not be 0 or 2. The thresholds and collection schedule for this metric are:</p> <ul style="list-style-type: none"> ■ Warning Threshold: NA ■ Critical Threshold: Down ■ Collection Schedule: Every 5 Minutes

Table 12–2 (Cont.) Available Server Metrics

Enterprise Manager Metric	JMX Parameter	Description
Number of Messages Sent to Federated Domain (S2S)	MessageXferPerS2SDomain	<p>The number of messages sent to the Instant Messaging Server federated domain.</p> <ul style="list-style-type: none"> ■ Warning Threshold: NA ■ Critical Threshold: NA ■ Collection Schedule: Every 10 Minutes
Active Users (server)	ActiveC2SCount	<p>The number of active Instant Messaging Server users.</p> <ul style="list-style-type: none"> ■ Warning Threshold: Greater than 80000 users ■ Critical Threshold: Greater than 100,000 users ■ Collection Schedule: Every 10 Minutes
Authenticated Users (server)	AuthenticatedUsersCount	<p>The total number of authentications passed since the server started.</p> <ul style="list-style-type: none"> ■ Warning Threshold: Greater than NA ■ Critical Threshold: Greater than NA ■ Collection Schedule: Every 10 Minutes
Average Number of Users per Chat Room	UsersPerMuc	<p>The average number of users per chat room.</p> <ul style="list-style-type: none"> ■ Warning Threshold: Greater than 50 users ■ Critical Threshold: Greater than 70 users ■ Collection Schedule: Every 10 Minutes
Average User Life	AverageC2SLifeInSec	<p>Average life of an authenticated client session.</p> <ul style="list-style-type: none"> ■ Warning Threshold: NA ■ Critical Threshold: NA ■ Collection Schedule: Every 10 Minutes
CPU Usage	ProcessCPUUsage	<p>The CPU percentage consumed by the Instant Messaging Server server process.</p> <ul style="list-style-type: none"> ■ Warning Threshold: Greater than 70% ■ Critical Threshold: Greater than 90% ■ Collection Schedule: Every 10 Minutes
Client Login Rate	C2SLoginRate	<p>The average number of client logins per second during the last data refresh interval.</p> <ul style="list-style-type: none"> ■ Warning Threshold: Greater than 10 logins per second ■ Critical Threshold: Greater than 15 logins per second ■ Collection Schedule: Every 10 Minutes
Free Disk Space in Disk Partition of the Instance Directory	FreeDiskSpace	<p>The amount of free disk space in megabytes in the instance partition.</p> <ul style="list-style-type: none"> ■ Warning Threshold: Less than 2048 MB ■ Critical Threshold: Less than 1024 MB ■ Collection Schedule: Every 10 Minutes

Table 12–2 (Cont.) Available Server Metrics

Enterprise Manager Metric	JMX Parameter	Description
Memory Usage	ProcessMemoryUsage	<p>The memory consumed in megabytes by the Instant Messaging Server server process.</p> <ul style="list-style-type: none"> ■ Warning Threshold: Greater than 10240 MB ■ Critical Threshold: Greater than 14336 MB ■ Collection Schedule: Every 10 Minutes
Message Transfer Rate	MessageXferPerSec	<p>The message transfer rate, as measured by the average number of messages per second during the last data refresh interval.</p> <ul style="list-style-type: none"> ■ Warning Threshold: Greater than 300 messages ■ Critical Threshold: Greater than 500 messages ■ Collection Schedule: Every 10 Minutes
Number of Active Chat Rooms	NumOfActiveMUC	<p>The number of active chat rooms.</p> <ul style="list-style-type: none"> ■ Warning Threshold: Greater than 1000 rooms ■ Critical Threshold: Greater than 1200 rooms ■ Collection Schedule: Every 10 Minutes
Number of File Transfers	FileXferCount	<p>The number of file transfers.</p> <ul style="list-style-type: none"> ■ Warning Threshold: NA ■ Critical Threshold: NA ■ Collection Schedule: Every 10 Minutes
Transaction Times	None	<p>The transaction times.</p> <ul style="list-style-type: none"> ■ Login ■ Roster fetch ■ Messages between two users
None	ConnectionStatusMap	The status of connections between peers.
None	PingFailureMap	A list of at most five time stamps that indicates the times when a ping failure happened with the respective peer.
None	MUCPersistenceState	The state of thread pool and queue in multiuser chat persistence thread.
None	ProcessQueueMap	The number of pending packets per process-queue.
None	WorkerQueueMap	The number of pending tasks per worker thread-pool queue.
None	DiskBackingCountMap	The rate at which outgoing packets are backed up on disk.
None	EmbryonicSessionsCount	The total number of idle unauthenticated client connections detected and closed by the server.
None	terminateClientSession	Given a full IID of a locally connected user, this operation terminates the session attached to the resource provided.
None	AverageMessageSize	The average number of characters in the body of message packets.

Table 12–2 (Cont.) Available Server Metrics

Enterprise Manager Metric	JMX Parameter	Description
None	ActiveMUCPerDomain	The number of active multiuser chat rooms per domain.
None	C2SFailedLoginRate	The users' failed login attempt rate per second for the last refresh timeout period.
None	AvgMUCParticipantsPerDomain	The average number of multiuser chat participants per domain.

Customizing Monitoring

You can customize the following aspects of monitoring:

- The thresholds for sending alerts and notifications (see ["Setting Thresholds on Monitored Metrics"](#))
- The notifications sent when a threshold is crossed (see ["Setting Notification Options"](#))
- The corrective actions taken when a threshold is crossed (see ["Adding Corrective Actions"](#))

Setting Thresholds on Monitored Metrics

To set thresholds for Instant Messaging Server data collection and metrics:

1. From the Enterprise Manager Cloud Control administration console, select a managed Instant Messaging Server target.
2. On the target's overview page, click the target name and from the **Monitoring** menu, select **Metric and Collections Settings**.
3. On the **Metric and Collections Settings** page, configure the monitoring thresholds as required for your environment by clicking the **Edit** icon adjacent to the metric in the **Metrics** table.
4. Click **OK** to save your changes.

Setting Notification Options

You can configure notification behavior by selecting **Setup** from the Enterprise Manager Cloud Control administration console then choosing from the options available in the **Notifications** sub menu. For detailed information on configuring notifications, see *Oracle Enterprise Manager Cloud Control Administrator's Guide*.

Adding Corrective Actions

You can add automatically executed corrective actions for when warning and critical thresholds are triggered. For information on corrective actions, see *Oracle Enterprise Manager Cloud Control Administrator's Guide*.

Instant Messaging Server Diagnostic Metrics

Instant Messaging server diagnostic metrics are turned off by default. You can enable them as needed to diagnose server behavior if a problem occurs. [Table 12–3](#) lists the available server diagnostic metrics.

Table 12–3 Available Server Diagnostic Metrics

JMX Parameter	Description
terminateClientSession	Given a full JID of a locally connected user, this operation terminates the session attached to the resource provided.
OutQueueMap	The number of pending packets per out-queue.
MUCPersistenceState	The state of thread pool and queue in multiuser chat persistence thread.
ProcessQueueMap	The number of pending packets per process-queue.
WorkerQueueMap	The number of pending tasks per worker thread-pool queue.
DiskBackingCountMap	The rate at which outgoing packets are backed up on disk.
Enabled	Displays whether the Server Diagnostics feature is enabled. The default is false .

Enabling and Disabling Instant Messaging Server Diagnostic Metrics

Caution: Enabling diagnostic metrics might impact system performance. Thus, only enable diagnostic metrics to help troubleshoot a problem.

To enable or disable Instant Messaging Server diagnostic metrics:

1. Connect to the Instant Messaging Server host through a JMX client, such as JConsole.
2. Navigate to the **Mbeans** tab and select **OCIM.Server**.
3. Click the **OCIM.Server**.
4. Click **ServerDiagnostics**.
5. Click **Operations**.
6. Click **toggleEnableState** to enable or disable diagnostic metrics.
7. Click **OCIM.Server**, then click **ServerDiagnostics** and select **Attributes** to verify the value of **Enabled** based on your choice to either enable or disable diagnostic metrics.

Overview of Monitoring the Instant Messaging Multiplexor

You can collect data and monitor Instant Messaging multiplexor activity. Examples of the types of data you can collect are:

- Total number of active client connections to the multiplexor
- Amount of memory consumed by the mux process (muxd)

You can use the collected data for purposes such as:

- Monitoring the health of the multiplexor and identifying quality of service issues
- Resource utilization and capacity planning

See ["Available Multiplexor Metrics"](#) for more information on the statistics that can be collected.

To monitor the Instant Messaging data collected you use a JMX client, such as JConsole. See "[Available Multiplexor Metrics](#)" for information on the JMX metrics available for monitoring. For information on accessing the collected metrics, see the Jconsole documentation at:

<http://docs.oracle.com/javase/7/docs/technotes/guides/management/jconsole.html>

Configuring Instant Messaging Multiplexor Monitoring

To configure data collection and monitoring, you must enable multiplexor monitoring and configure JMX-based server monitoring, the JVM, and the JAAS. For more information on JMX and JAAS settings and configuration files, see the JMX documentation at:

<http://docs.oracle.com/javase/7/docs/technotes/guides/management/agent.html>

Steps for Configuring Data Collection and Monitoring

All configuration in the following steps is carried out through the **imconfutil** command.

1. Log in to the multiplexor host as **root**.
2. Enable monitoring by setting the **iim_mux.monitor.enable** property to **true**:

```
imconfutil set-prop iim_mux.monitor.enable=true -c InstantMessaging_
home/config/iim.conf.xml
```

3. Set the user name and password for the server to use in JAAS authentication:

```
imconfutil set-prop iim_mux.admin.user=UserID -c InstantMessaging_
home/config/iim.conf.xml
imconfutil set-prop iim_mux.admin.password=Password -c InstantMessaging_
home/config/iim.conf.xml
```

You must also assign the user name you enter **readwrite** permission in the JMX access-control file.

4. Set the interval, in seconds, at which you want monitored data to be refreshed. The default interval is 30 seconds. To configure an alternative interval:

- Set the **iim_mux.monitor.refreshtimeout** property as in the following example:

```
imconfutil set-prop iim_mux.monitor.refreshtimeout=60 -c InstantMessaging_
home/config/iim.conf.xml
```

- If you are using a JMX client, you can use the **refreshNow** operation in the JMX Statistics MBean for an immediate refresh.
5. Configure JVM and JAAS properties for monitoring.

The Instant Messaging Server stores all JVM and JAAS properties in a single configuration property that is passed to the server process. [Table 12-4](#) lists the properties you must set.

Table 12–4 Property Settings for Configuring JMX and JAAS for Server Monitoring

Property	Description
-Dcom.sun.management.jmxremote	Enables remote monitoring.
-Dcom.sun.management.jmxremote.port=<i>port_number</i>	Sets the JMX remote access port for the client connection.
-Dcom.sun.management.jmxremote.local.only=false	Allows remote JMX access.
-Dcom.sun.management.jmxremote.ssl=true	true enables SSL for the JMX connection, false disables it.
-Dcom.sun.management.jmxremote.authenticate=true	Enables JAAS authentication.
-Djava.security.auth.login.config=<i>InstantMessaging_home/config/jaas_login_config_file</i>	Specifies the location of the JAAS login configuration file to use for authentication. For more information, see the JAAS documentation at: http://docs.oracle.com/javase/7/docs/technotes/guides/security/jgss/tutorials/LoginConfigFile.html
-Dcom.sun.management.jmxremote.login.config=<i>JAAS_Login</i>	Specifies the name of JAAS login entry in the JAAS login configuration file. The name of the entry must match the name you enter for -Dcom.sun.management.jmxremote.login.config with the imconfutil command.
-Dcom.sun.management.jmxremote.access.file=<i>InstantMessaging_home/config/jmxaccess</i>	Specifies the absolute path to the JMX access file. You can create a jmxaccess file based on the JMX-access template file located at: <i>JDK_InstallDir/jre/lib/management/jmxremote.access</i> . In configuring the jmxaccess file, the name of the user specified in controlRole must be same as the user name you entered for the server to use in JAAS authentication.
-Dcom.sun.management.jmxremote.ssl=true	true enables SSL for the JMX connection, false disables it. If you enable SSL, supply the keystore and password as well.
-Djavax.net.ssl.keyStore	Specifies the location of the keystore.
-Djavax.net.ssl.keyStorePassword	Specifies the location of the keystore password.
-Dcom.sun.management.jmxremote.ssl.needs.client.auth=false	Disables mutual SSL authentication.

You must set all of the properties in a single **imconfutil** command. The **imconfutil** command in the following example sets the JVM and JAAS properties required for monitoring.

```
imconfutil set-prop iim_
mux.jvm.options="-Dcom.sun.management.jmxremote.access.file=/opt/sun/comms/im/c
onfig/jmxaccess -Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=9010
-Dcom.sun.management.jmxremote.local.only=false
-Dcom.sun.management.jmxremote.authenticate=true
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.login.config=Login1
-Djava.security.auth.login.config=/opt/sun/comms/im/config/jaasconfig" -c
InstantMessaging_home/config/iim.conf.xml
```

6. Add an entry to the JAAS login configuration file that contains the **com.oracle.im.stat.AdminLoginModule** implementation of **LoginModule**. For information, see the topic on configuring JAAS login entries at:

<http://docs.oracle.com/javase/7/docs/technotes/guides/security/jaas/JAASRefGuide.html#AppendixB>

The name of the entry must match the name you enter for
-Dcom.sun.management.jmxremote.login.config with the **imconfutil** command.

Available Multiplexor Metrics

Table 12–5 lists the available multiplexor metrics.

Table 12–5 Available Multiplexor Metrics

JMX Parameter	Description
ActiveConnectionCount	The active client connection to the multiplexor.
C2SConnectionRate	The client-to-multiplexor connection rate per second.
FreeDiskSpace	The free disk space in megabytes available on the disk partition of the instance directory for the multiplexor.
ProcessCPUUsage	The CPU usage of the multiplexor process.
ProcessMemoryUsage	The memory usage (heap plus non-heap) in megabytes by multiplexor process.

Instant Messaging Multiplexor Diagnostic Metrics

Instant Messaging multiplexor diagnostic metrics are turned off by default. You can enable them as needed to diagnose multiplexor behavior if a problem occurs.

Table 12–6 lists the available multiplexor diagnostic metrics.

Table 12–6 Available Multiplexor Diagnostic Metrics

JMX Parameter	Description
WorkerQueueMap	The state of the worker threadpool's queue.
DiskBackingCountMap	The number of backed-up packets in disks per connection.
Enabled	Displays whether the MUX Diagnostics feature is enabled. The default is false .

Enabling and Disabling Instant Messaging Server Diagnostic Metrics

Caution: Enabling diagnostic metrics might impact system performance. Thus, only enable diagnostic metrics to help troubleshoot a problem.

To enable or disable Instant Messaging multiplexor diagnostic metrics:

1. Connect to the Instant Messaging multiplexor host through a JMX client, such as JConsole.
2. Navigate to the **Mbeans** tab and select **OCIM.Multiplexor**.
3. Click the **OCIM.Multiplexor**.
4. Click **MuxDiagnostics**.
5. Click **Operations**.
6. Click **toggleEnableState** to enable or disable diagnostic metrics.

7. Click **OCIM.Multiplexor**, then click **MuxDiagnostics** and select **Attributes** to verify the value of **Enabled** based on your choice to either enable or disable diagnostic metrics.

Troubleshooting Instant Messaging Server

This chapter describes common problems that might occur during installation and deployment of Oracle Communications Instant Messaging Server. It also provides an overview of the watchdog process.

Troubleshooting and Monitoring Instant Messaging Server Overview

The log information generated by the various system components on their operation can be extremely useful when trying to isolate or troubleshoot a problem. In addition, you can use the monitoring framework agent to monitor the general health of Instant Messaging Server processes to help prevent problems before they occur, assess usage levels to help you scale your deployment, and to prevent downtime.

See ["Managing Logging for Instant Messaging Server"](#) for details and more information on managing server, multiplexor, watchdog, Calendar agent, and client logging, and for default log file locations.

General Troubleshooting

This section describes general problems and solutions.

Cannot Forward Mail to Offline Users

By default, Instant Messaging Server uses the **mail** attribute to determine the email address to which it forwards instant messages when a recipient is offline. If your directory does not use the **mail** attribute for email addresses, you must configure Instant Messaging Server to use the same attribute as your directory.

Configuring the Attribute Used for User Email Addresses

Use the **imconfutil** command to change the value of the **iim_ldap.usermailattr** property to the attribute that your directory uses to contain email addresses in user entries.

Calendar Pop-up Reminders Do Not Work

If Calendar pop-ups are not being delivered as expected, you can troubleshoot the configuration as described in this section. See ["Using Calendar Pop-up Reminders"](#) for instructions on setting up Calendar pop-ups.

The most common error in Calendar pop-up configuration is incorrectly entered property names in the configuration files. This includes typos and misspelled property names. Ensure that you have correctly entered all of the configuration properties and

values in the **iim.conf.xml** and **ics.conf** files. If you have already configured pop-ups, use "[JMQ Properties](#)" to compare your entries with the required properties.

If your Instant Messaging Server and Calendar Server configuration files are correct, but pop-ups are still not arriving as expected, ensure the Calendar client and Instant Messaging client are configured correctly.

If you received the email alert, but not the Calendar pop-up, and you are sure that you have configured both servers and clients correctly, check the **xmppd.log** for further information. You might need to set this log to a more verbose setting, for example **DEBUG**. See "[Managing Logging for Instant Messaging Server](#)" for instructions on changing the log level.

Connection Refused or Timed Out

The following are the possible causes for this problem:

- Either the Instant Messaging server or the multiplexor is not running.
- Incorrect multiplexor host or port names used in the Applet descriptor file **.jnlp** or **.html**.
- Different SSL settings used between instant messaging client and the multiplexor.
- Client and server version mismatch.

Where to get diagnostic information:

- Instant Messaging server and multiplexor log files

Authentication Errors

The following are the possible causes for this problem:

- Problems while accessing the LDAP server, such as the LDAP server is not running, or a provisioning error, such as a schema violation, has occurred
- End user not found
- Invalid credentials

Where to get diagnostic information:

- Instant Messaging server, Identity authentication, and LDAP log files.

Instant Messaging Server Content is not Archived

The following are the possible causes for this problem:

- Content is actually archived but the end user has insufficient rights to access it.
- The content has not yet been committed to the database.
- The archive provider has been disabled in the Instant Messaging server.

You can get diagnostic information from the Instant Messaging server log files.

Server-to-Server Communication Fails to Start

The following are the possible causes for this problem:

- Incorrect server identification
- Mismatch in the SSL settings

You can get diagnostic information from the Instant Messaging server log file for both servers.

Troubleshooting Instant Messaging Server and LDAP

The following LDAP issues might arise in a given deployment. Change the LDAP properties in the **iim.conf.xml** file accordingly.

Using a Directory That Does not Permit Anonymous Bind

By default, Instant Messaging Server performs an anonymous search of the LDAP directory. However, it is common for sites to prevent anonymous searches in their directory so that any random person cannot do a search and retrieve all the information. If your site's directory is configured to prevent such anonymous searches, and you did not provide bind credentials during post-installation configuration, you must configure Instant Messaging Server with a user ID and password it can use to bind and perform searches.

Use the **iim_ldap.usergroupbinddn** and **iim_ldap.usergroupbindcred** properties to configure the necessary credentials.

Configuring Bind Credentials for Instant Messaging Server

To configure bind credentials for Instant Messaging Server, use the **imconfutil** command to set the **iim_ldap.usergroupbinddn** and **iim_ldap.usergroupbindcred** properties:

```
imconfutil set-prop -c InstantMessaging_home/config/iim.conf.xml iim_ldap.usergroupbinddn="cn=Directory Manager" iim_ldap.usergroupbindcred=password
```

Changing the Attribute Used to Display Contact Names

To change the attribute used to display contact names, use the **imconfutil** command to set the **iim_ldap.userdisplay** and **iim_ldap.groupdisplay** properties:

```
imconfutil set-prop -c InstantMessaging_home/config/iim.conf.xml iim_ldap.userdisplay=sn iim_ldap.groupdisplay=sn
```

Searching the Directory by Using Wildcards

If your directory is indexed to allow the use of wildcards, and you want to be able to use wildcards while searching for contact names, you must configure Instant Messaging Server to enable wildcard searches. However, enabling wildcard searches can impact performance unless User IDs are indexed for substring search.

- Use the **imconfutil** command to set the **iim_ldap.usergroupbynamefilter** attribute. This property specifies the LDAP search string used when searching for users or groups. Provide the attribute value in standard LDAP filter syntax. You can modify it to allow more complex searches. See your Directory Server documentation for more information on modifying search strings.

Using Nonstandard Objectclasses for Users and Groups

If your directory uses nonstandard object classes to define users and groups you must change the appropriate **iim_ldap.*** properties, replacing **inetorgperson** and **groupofuniquenames** with your values.

See [Table 28–2, "LDAP, User Registration, and Source Configuration Properties"](#) for a list of LDAP properties.

Changing the Object Classes Used to Specify Users and Groups

To change the object classes used to specify users and groups:

1. Use the **imconfutil** command to change the **iim_ldap.*** properties.
2. Search for and replace **inetorgperson** with the object class used to define users in your directory.
3. Search for and replace **groupofuniqueNames** with the object class used to define groups in your directory.

Using an Attribute Other than uid for User Authentication

If your directory does not use the **uid** attribute for user authentication, you must configure Instant Messaging Server with the attribute used by your directory. By default, Instant Messaging Server uses **uid**. You also need to change each filter property that contains **uid** in its value.

Use the **iim_ldap.loginfilter** property to specify which attribute to use for user authentication.

Changing the Attribute Used for User Authentication

Use the **imconfutil** command to set the **uid** with the attribute you want to use for user authentication in the **iim_ldap.loginfilter** and **iim_ldap.usergroupbyidsearchfilter** properties.

Using an Attribute Other than uid for User IDs

If your directory does not use the **uid** attribute for user IDs, you must configure Instant Messaging Server with the attribute used by your directory. By default, Instant Messaging Server uses **uid**. In addition, you should index the attribute in the directory to help offset any performance degradation caused by searching on unindexed attributes.

Use the **iim_ldap.useruidattr** property to specify which attribute to use for user IDs.

Changing the Attribute Used for User IDs

To change the user ID attribute:

1. Use the **imconfutil** command to set the attribute you want to use for user IDs as the value for the **iim_ldap.useruidattr** property. For example, to use the **loginname** attribute:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml iim_ldap.useruidattr=loginname
```

2. Add the index directive to the indexing rules in LDAP:

```
index loginname eq
```

Troubleshooting Connectivity Issues in a Multi-Node Deployment (Server Pool)

If you are receiving errors where presence status is not being shared between servers in a server pool:

- Ensure that the nodes are configured correctly to enable server-to-server communication. See ["About Server Pooling"](#) for more information.
- Check for server-to-server session establishment errors in the log file.

Managing the Watchdog Process

The watchdog process monitors the server and multiplexor components and attempts to restart a component if it determines that the component is not running.

For the server, the watchdog determines whether the server is running by periodically attempting to make a connection, either directly to the server or through the multiplexor, based on the current configuration of the server. The watchdog tries to poll the server's operational status and if it cannot determine the status, it then tries to make a connection to the server. If both operations fail, the watchdog stops and then restarts the server.

Before you use the watchdog, verify that it is enabled and running using the **imadmin status** command. By default, the watchdog is enabled and running when you install Instant Messaging Server.

See ["imadmin Command Reference"](#) for more information about the **imadmin** command.

Determining the Status of the Watchdog

Use the **imadmin** command to check the status of the watchdog:

1. Change to the directory that contains the **imadmin** command.

```
cd InstantMessaging_home/sbin
```

2. Run **imadmin status**:

```
imadmin status watchdog
```

The **imadmin** command returns the current status of the watchdog.

Enabling and Disabling the Watchdog

By default, the watchdog is enabled when you install Instant Messaging Server. You can disable or enable the watchdog by setting a configuration property in the **iim.conf.xml** file.

1. Use the **imconfutil** command to either enable or disable the watchdog by setting the **iim_wd.enable** property as follows:

To enable the watchdog:

```
iim_wd.enable=true
```

To disable the watchdog:

```
iim_wd.enable=false
```

2. Refresh the Instant Messaging Server configuration:

```
imadmin refresh
```

Managing Logging for the Watchdog

You manage logging for the watchdog the same way you manage logging for the server, multiplexor, and the Calendar agent. The watchdog log file is saved as *InstantMessaging_database/log/iim_wd.log*.

See "[Managing Logging for Instant Messaging Server](#)" for more information on setting logging levels for all Instant Messaging Server components including the watchdog.

Managing Logging for Instant Messaging Server

This chapter provides information about Oracle Communications Instant Messaging Server logging. Instant Messaging Server implements log4j and includes logging for the server, the calendar agent, the watchdog, the multiplexor, the gateway connector, the XMPP/HTTP gateway (see "[Configuring the XMPP/HTTP Gateway](#)"), and other features. For information about log4j, see:

<http://logging.apache.org>

Instant Messaging Server Log File Location

You specify the location of the log files when you run the **configure** utility after installing Instant Messaging Server. Typically, log files are stored in the *InstantMessaging_runtime/log* directory. See "[Configuration File and Directory Structure Overview](#)" for information on the location of *InstantMessaging_runtime*.

If you are using log4j for log file generation in your deployment, the logger will also use the directory you specify during configuration as the base directory in which to store log4j logs.

Instant Messaging Server Component Logging Levels

The level or priority of maintaining the error log defines how detailed, or verbose, the log should be. A higher priority level implies less details as only events of high priority (high severity) are recorded in the log file. In contrast a lower priority level implies greater details as more events are recorded in the log file.

You can set the logging level separately for each component.

[Table 14-1](#) describes the logging levels for the components. These logging levels are a subset of the levels defined by the UNIX **syslog** facility.

Table 14-1 Logging Levels for Instant Messaging Server Components

Level	Description
FATAL	This priority level records minimum logging details in the log file. A log record is added to the log file whenever a severe problem or critical condition occurs. If a FATAL problem occurs, the application might stop functioning.
ERROR	A log record is added to the log file whenever a recoverable software error condition occurs or a network failure is detected. For example, when the server fails to connect to a client or to another server.

Table 14–1 (Cont.) Logging Levels for Instant Messaging Server Components

Level	Description
WARNING	A log record is added to the log file whenever a user error is detected. For example, when the server cannot understand the communication sent by the client.
INFO	A log record is added to the log file whenever a significant action takes place. For example, when an end user successfully logs in or logs out.
DEBUG	The tasks are recorded in the log file. This information is useful for debugging purposes only. Each event with individual steps, within each process or task, is written in the log file. The information recorded helps the end user identify the problems while debugging the application.

When you select a particular logging level, events corresponding to that level and to all higher and less verbose levels are logged.

INFO is the default level for the server. **ERROR** is the default level for the multiplexor, Calendar agent, and watchdog log files.

Managing Instant Messaging Server Logging by Using log4j

This section describes using the log4j logger to generate log files for Instant Messaging Server.

When you install Instant Messaging Server, a template file (**log4j.conf.template**) for the log4j configuration file is installed into the *InstantMessaging_home/lib* directory.

When you run the **configure** utility after installation, the template is used to create the log4j configuration file (**log4j.conf**) in the *InstantMessaging_cfg* directory. This configuration file is used to determine where to store log files generated by log4j, the logging level to use for various components, the output syntax, and to determine what log files to generate.

The logging levels described in "[Instant Messaging Server Component Logging Levels](#)" are used by the log4j logger.

For more information about log4j, and instructions on configuring aspects of log files, such as size, number of backups, and so on, see Apache Logging Services at:

<http://logging.apache.org>

Instant Messaging Server Log4j Configuration File (log4j.conf) Location

You can change the location of the log4j configuration file, **log4j.conf**, by using the **imconfutil** command to modify the **iim.log4j.config** configuration property. If you do not specify a value for this parameter, the logger checks the *InstantMessaging_cfg* directory. If the logger does not find the log4j configuration file in that directory, it uses the logging configuration properties to generate non-log4j style logs.

See "[Configuration File and Directory Structure Overview](#)" for information on locating *InstantMessaging_cfg*.

Instant Messaging Server Log4j Log File Syntax

The **configure** utility generates the log4j configuration file (**log4j.conf**) based on the content of the log4j configuration file template (**log4j.conf.template**). In the template:

- `${logdir}` corresponds to the directory in which you want to store log files, as specified during configuration. See ["Instant Messaging Server Log File Location"](#) for more information.
- Appenders are identified by IDs of the form `Ax`, where `x` is an integer, for example, `log4j.appender.A2.maxFileSize=5mb`.
- There are separate sections for configuring logging for different types of data. Each section starts with an initial property of the form `log4j.logger.identifier`, for example: `log4j.logger.agent-calendar`. The following identifiers for types of logging are available:
 - `xmppd` - Generates `xmppd.log`, which contains logging information for the server.
 - `iim_wd` - Generates `wd.log`, which contains information for the watchdog.
 - `xmppd.xfer` - Generates `xfer.log`, which is only for XMPP traffic.
 - `agent-calendar` - Generates logging information for the Calendar agent.
 - `net.outer_planes.jso.BasicStream` - Generates `jso.log`, which contains information for Jabber stream objects.
 - `muxd` - Generates `muxd.log`, which contains logging information for the multiplexor.
 - `smppbind` - Generates `smppbind.log`, which contains logging information for the SMS Gateway.
 - `router` - Generates `relay.log`, which contains logging information for shoal relay.

Log4j Log Levels for Instant Messaging Server Components

The log4j logger uses the same logging levels described for configuration property-based logging mechanism in ["Instant Messaging Server Component Logging Levels"](#).

Specifying the Location of the log4j Configuration File (log4j.conf)

To specify the location of the `log4j.conf` file:

1. Use the `imconfutil` command to set the `iim.log4j.config` configuration property to the path in which you want the logger to look for `log4j.conf`.

On Solaris OS:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop
iim.log4j.config=/etc/opt/sun/comms/im/default/config/log4j.conf
```

On Linux:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop
iim.log4j.config=/etc/opt/sun/comms/im/default/config/log4j.conf
```

2. Refresh the server.

```
imadmin refresh
```

Enabling or Disabling log4j Logging for a Component

By default, log4j logging is used for all components for which logging information is generated.

1. To disable log4j logging, set the logging level for the component to **OFF** in both **log4j.conf** and **log4j.conf.template**.

See ["Setting log4j Log Levels"](#) for more information.

2. To enable log4j logging, set the logging level for the component to any logging level other than **OFF** in both the **log4j.conf** and **log4j.conf.template** files.

Setting log4j Log Levels

You can set log levels by modifying either the template or the log configuration file. However, if you only modify the configuration file, any changes you make are overwritten the next time you run configure. To prevent overwriting changes, you should make your changes to both the configuration file and the template.

1. Open **log4j.conf.template**.

By default, this file is stored in the *InstantMessaging_home/lib* directory.

2. For each component, specify the logging level you want to use.

For example, to set the log level for the server:

```
log4j.logger.xmppd=log-level
```

where *log-level* is one of **FATAL**, **ERROR**, **WARNING**, **INFO**, or **DEBUG**.

See [Table 14-1, "Logging Levels for Instant Messaging Server Components"](#) for detailed information on each of these logging levels.

3. Save and close the **log4j.conf.template** file.
4. Repeat the procedure for the configuration file **log4j.conf**.

Specifying the Maximum log4j Log File Size

You can set log levels by modifying either the template or the log configuration file. However, if you only modify the configuration file, any changes you make will be overwritten the next time you run configure. To prevent overwriting changes, you should make your changes to both the configuration file and the template.

1. Open the **log4j.conf.template** file.

By default, this file is stored in the *InstantMessaging_home/lib* directory.

2. For each component, specify the maximum size for the component's log file.

For example, to set the size for the server log file:

```
log4j.appender.A1.maxFileSize=max-logfile-size
```

where **A1** is the default appender ID for the server, **max-logfile-size** is in MB, for example 5MB.

3. Repeat the procedure for the configuration file **log4j.conf**.

Part II

Configuring Gateways, Protocols, and Features

Part II contains the following chapters:

- [Configuring Hosted Domain Support](#)
- [Federating Instant Messaging Server with External Servers](#)
- [Configuring the XMPP WebSocket Gateway](#)
- [Configuring the XMPP/HTTP Gateway](#)
- [Configuring External Service Discovery](#)
- [Configuring the SIP Gateway](#)
- [Configuring the SMS Gateway](#)
- [Using Calendar Pop-up Reminders](#)
- [Configuring the Instant Messaging Server Calendar Agent](#)
- [Displaying Availability Based on Calendar Entries](#)
- [Using the Web Presence API](#)
- [Configuring the Instant Messaging Server Web Presence API](#)

Configuring Hosted Domain Support

This chapter describes how to configure hosted domains for Oracle Communications Instant Messaging Server.

About Instant Messaging Server Hosted Domains

Instant Messaging Server provides support for hosted domains. In a hosted domain installation, each domain shares the same instance of Instant Messaging Server that enables multiple domains to exist on a single server. Each hosted domain has a name space that can contain unique users, groups, resources, preferences, and attributes.

Communication between hosted domains is disabled by default. To enable cross domain communication, see ["Enabling Communication Between Hosted Domains"](#).

Setting Up Schema 1 and Schema 2 for Instant Messaging Server Hosted Domains

Instant Messaging Server supports two schema versions: **Schema 1** and **Schema 2**. This section describes the steps to set up the schema for hosted domains.

Schema 1 Structure

The directory structure of **Schema 1** includes two trees for domain management: the organization tree and the domain component (DC) tree. For example, for domain **xyz.abc.com**, the tree structure is as follows:

```
A, dc tree: o=internet // dc tree root suffix
dc=com
dc=abc
dc=xyz // domain node
```

The domain should contain the following attributes:

- **objectclass=inetDomain**
- **inetDomainBaseDn=o=xyz.abc.com**
- **dc=xyz,dc=abc, dc=com**

inetDomainBaseDn is a mandatory attribute for the **inetDomain** object class. You should also specify the status of the **inetDomainStatus** attribute as active.

o=xyz.abc.com, dc=xyz,dc=abc,dc=com is the domain name of organization in the organization tree that contains the users for the domain **xyz.abc.com**.

Configuring Instant Messaging Server for Schema 1

To configure Instant Messaging Server for Schema 1:

- Run the **imconfutil** command to set the necessary properties.

```
imconfutil set-prop -c path iim_server.usesso=0 iim.policy.modules=iim_ldap_
schema1 iim.userprops.store=ldap iim_ldap.schema1.domain_config_root=value
```

where:

path is the full directory path to and including **iim.conf.xml**, for example, **/opt/sun/comms/im/config/iim.conf.xml**.

xml.iim_ldap.schema1.domain_config_root is the DC tree root suffix, for example, **o=internet**.

Schema 2 Structure

Schema 2 has only the DC as the **config** root. **Schema 2** has the following tree structure:

```
B, Organization tree: dc=xyz,dc=abc,dc=com // Base dn for users/groups
o=xyz.abc.com
ou=people // Users are under this node
```

Configuring Instant Messaging Server for Schema 2

To configure Instant Messaging Server for Schema 2:

- Run the **imconfutil** command to set the necessary properties.

```
imconfutil set-prop -c path iim_server.usesso=0 iim.policy.modules=iim_ldap_
schema2 iim.userprops.store=ldap iim_ldap.schema2.domain_config_root=value
```

where:

path is the full directory path to and including **iim.conf.xml**, for example, **/opt/sun/comms/im/config/iim.conf.xml**.

iim_ldap.schema2.domain_config_root is the DC tree root suffix, for example, **dc=red,dc=example,dc=com**.

If the default value of the **iim.policy.modules** property is **iim_ldap**, the users under the non-default domain cannot be searched. Users cannot log in to Instant Messaging Server. Instant Messaging Server, in this case, does not go through the DC tree to find the value of the **inetDomainBaseDn** attribute. The server uses the value of the **iim_ldap.searchbase** attribute to search users who exist in the default domain. You can specify the default domain by using the **iim_server.domainname** attribute.

iim_ldap.schema2.domain_filter specifies the object class of the domain node. The default value is **inetDomain**.

Note: Instant Messaging Server does not provide a tool to create these topologies.

Instant Messaging Server Cross Domain Searches

Cross domain search functionality enables users in one domain to search for users and groups in other domains. The search is enabled for contacts and conferences.

Enabling Instant Messaging Server Cross Domain Searches

To enable Instant Messaging Server to search across domains:

1. Run the **imconfutil** command to set the necessary properties.

```
imconfutil set-prop -c path iim_server.discofilter.principal.any=true iim_
server.discofilter.conference.any=true iim_server.discofilter.domains.any=true
```

where:

path is the full directory path to and including **iim.conf.xml**, for example, **/opt/sun/comms/im/config/iim.conf.xml**.

2. Add the following property, which loads the specified domains into the server memory upon server startup.

```
imconfutil set-prop -c path iim_server.default_domains=value
```

where:

iim_server.default_domains is the domain, or comma-separated list of domains, on the server.

About Hosted Domains Communication

When domains reside in different networks, are hosted, or are managed by different service providers, you typically use XMPP server-to-server (S2S) inter-domain federation. Instant Messaging Server's S2S federation provides ways to enable and restrict communication between domains by using blacklists and whitelists. For domains that are hosted and served by the same Instant Messaging Server, you use a different method to enable or disable communication. Communication between hosted domains is disabled by default.

Enabling Communication Between Hosted Domains

To enable communication between hosted domains, set the **iim_server.hosteddomains.allowcrossdomainsaccess** property to **true**:

```
./imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.hosteddomains.allowcrossdomainsaccess=true
```

When you enable communication between hosted domains, set the **iim_server.hosteddomains.activelist** property to **whitelist**, **blacklist**, or **none**, depending on how you want to allow communication.

Disabling Communication Between Hosted Domains

To disable communication between hosted domains, set the **iim_server.hosteddomains.allowcrossdomainsaccess** property to **false**:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.hosteddomains.allowcrossdomainsaccess=false
```

Whitelisting and Blacklisting Domains for Hosted Domain Communication

To add whitelist domains for communication between hosted domains, use the **iim_server.hosteddomains.whitelist** property:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.hosteddomains.activelist=whitelist iim_
```

```
server.hosteddomains.whitelist=domain1, domain2
```

To add blacklist domains for communication between hosted domains, use the **iim_server.hosteddomains.blacklist** property:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_  
server.hosteddomains.activelist=blacklist iim_  
server.hosteddomains.blacklist=domain1, domain2
```

Federating Instant Messaging Server with External Servers

This chapter describes how to federate Oracle Communications Instant Messaging Server deployments to enable communications between users on different servers.

Overview of Federating Multiple Instant Messaging Servers

Instant Messaging Server supports inter-domain communication through *federation*. You can enable federation between two XMPP servers serving two different domains or between an XMPP server and a SIP/SIMPLE (Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions/Session Initiation Protocol) server serving two different domains through the SIP Gateway. This solution enables users on XMPP or SIP/SIMPLE networks to be able to communicate with each other.

When federating between multiple Instant Messaging servers, end users from different servers can communicate with each other, and use conference rooms on other domains, based on their access privileges.

In an LDAP-only deployment, the two servers should reside in different domains. For enabling communication between multiple Instant Messaging servers in your network, you must perform the configuration described in this chapter.

When federating between Instant Messaging Server and a server on a SIP/SIMPLE network, end users from different servers should be able to subscribe to each other's presence, send one-to-one chat messages, and enforce the privacy preferences. For enabling communication between Instant Messaging servers and SIP/SIMPLE servers, see "[Configuring the SIP Gateway](#)".

If both XMPP and SIP open federation are enabled, and a domain has both XMPP and SIP servers, XMPP federation is the preferred option.

Securing Server-to-Server Communication

Secure your server-to-server communication by using transport-layer security (TLS). This is required to prevent-third party infringement of security when data is exchanged between two servers. This precaution is extremely desirable in the case where the link between the two servers uses the public Internet. See *Instant Messaging Server Security Guide* for information on how to configure TLS.

Note: You can use the server-to-server federation only if the servers are using the same protocol. Instant Messaging Server uses the XMPP protocol. Thus, you can federate a server-to-server communication with GTalk or Openfire servers. In addition, Instant Messaging Server provides support for a user on an XMPP network to communicate with a user on a SIP/SIMPLE network through the SIP gateway.

Configuring Federated Communication Between Instant Messaging Servers

This section describes how to enable federated communication, either between two XMPP servers or between XMPP servers and SIP/SIMPLE servers. Both federated XMPP and SIP deployments require that you set the **iim_server.federation.policy** to **OPEN**. See ["About the SIP Gateway"](#) for more information on SIP. When you enable federation between XMPP and SIP/SIMPLE servers, additional steps are required, as described in ["Configuring the SIP Gateway"](#).

[Table 16–1](#) lists the federation configuration properties.

Table 16–1 Federation Configuration Properties

Property	Default Value	Description
iim_server.federation.policy	CLOSED	Specifies whether the server is allowed to federate with all other XMPP or SIP/SIMPLE servers, where OPEN indicates that it is allowed and CLOSED indicates that it is not allowed.
iim_server.federation.exceptions	None	Specifies a blacklist of domains where federation for XMPP or SIP/SIMPLE servers is denied. When iim_server.federation.policy=CLOSED and this property is set, the result is whitelisted domains where federation for XMPP or SIP/SIMPLE servers is allowed.

Federation Examples

You use the **imconfutil** command to set the federation configuration properties.

- To enable federation, set **iim_server.federation.policy=OPEN**.

When this property is set, any domain is able to federate with this Instant Messaging Server host.

- To disable federation, set **iim_server.federation.policy=CLOSED**.

When this property is set, no domain is be able to federate with this Instant Messaging Server host.

- To achieve open federation, but with a few domains blacklisted, set **iim_server.federation.policy=OPEN** and **iim_server.federation.exceptions=domain1.com,domain2.com**.

In this example, federation is allowed for any domain except *domain1.com* and *domain2.com*.

- To achieve federation with only a small whitelist of domains, set **iim_server.federation.policy=CLOSED** and **iim_server.federation.exceptions=domain1.com,domain2.com**.

In this example, federation is allowed only for the *domain1.com* and *domain2.com* examples, and no other domains.

Note: The domains in the exception list can be XMPP domains or SIP domains. See ["Configuring the SIP Gateway"](#) for more information on SIP.

To have the configuration change take effect, restart Instant Messaging Server:

```
imadmin refresh server
```

Configuring DNS for XMPP Federation

In a federated deployment, the XMPP server must publish **_xmpp-server** DNS SRV records for supported XMPP domains in the DNS server.

[Table 16–2](#) lists the DNS SRV records required for instant messaging and conference services.

Table 16–2 DNS SRV Records for XMPP Server

service._proto.name	TTL	Class	Type	Priority	Weight	Port	Target
xmpp-server._tcp.example.com	86400	IN	SRV	5	0	5269	xmpp.example.com
_xmpp-server._tcp.muc.example.com	86400	IN	SRV	5	0	5269	xmpp.example.com

The TTL (time-to-live) value informs the other server how long to cache the DNS entry. “muc” is a configurable service name for the conference service. **xmpp.example.com** is either the fully qualified domain name or IP address of the XMPP server.

To verify that the SRV record is added correctly in DNS entries, run the following **nslookup** command:

```
nslookup -type=srv _xmpp-server._tcp.example.com
```

If you are using SIP/SIMPLE Federation Gateway Services, then the SIP Gateway must publish **_xmpp-server** DNS SRV records for the SIP federated domains.

[Table 16–3](#) lists the DNS SRV record required for the SIP Gateway.

Table 16–3 DNS SRV Record for SIP Gateway

service._proto.name	TTL	Class	Type	Priority	Weight	Port	Target
xmpp-server._tcp.example.com	86400	IN	SRV	5	0	5269	sfs.example.com

sfs.example.com is either the fully qualified domain name or the IP address of the Oracle Communications Converged Application Server.

Note: The Instant Messaging Server SIP Gateway supports XMPP federation with single XMPP and SIP domains.

Configuring DNS for SIP Federation

If you are using SIP/SIMPLE Federation Gateway Services, then the SIP Gateway must discover SIP domains for Instant Messaging and presence federation.

[Table 16–4](#) lists the DNS SRV records required for Oracle Communications Converged Application Server to discover the SIP/SIMPLE server.

Table 16–4 *DNS SRV Records for SIP/SIMPLE Server*

service._proto.name	TTL	Class	Type	Priority	Weight	Port	Target
_sip._udp.example.com	86400	IN	SRV	5	0	5060	sip.example.com
_sips._udp.example.com	86400	IN	SRV	5	0	5061	sip.example.com
_sips._tcp.example.com	86400	IN	SRV	5	0	5060	sip.example.com
_sips._tcps.example.com	86400	IN	SRV	5	0	5061	sip.example.com

sip.example.com is either the fully qualified domain name or the IP address of the SIP/SIMPLE server.

To verify that the SRV record is added correctly in DNS entries, run the following **nslookup** command:

```
nslookup -type=srv _sip._udp.example.com
```

Configuring the XMPP WebSocket Gateway

This chapter describes the XMPP WebSocket Gateway feature and how to configure it for Oracle Communications Instant Messaging Server.

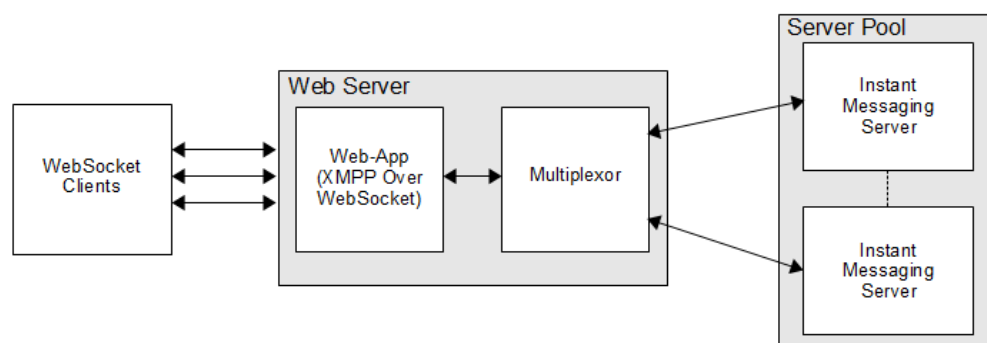
About the XMPP WebSocket Gateway

The XMPP WebSocket Gateway enables the Instant Messaging Server to support the WebSocket protocol for XMPP. XMPP over WebSocket is an efficient solution for HTTP access of Instant Messaging Server other than HTTPBind. The Instant Messaging multiplexor enables scaling of TCP client connections and also provides failover to other Instant Messaging Server nodes. These existing capabilities of the multiplexor can be reused when WebSocket client connections are interfaced with it.

A web container is used to create WebSocket end-points. These end-points in turn use the multiplexor to connect with the Instant Messaging Server. This is in contrast to HTTPBind or PresenceAPI components, which use component protocol to connect to Instant Messaging Server. The XMPP WebSocket Gateway uses internal multiplexor protocols to connect with Instant Messaging Server. The gateway implements XMPP over WebSocket, as described by RFC 7395.

Figure 17-1 shows that WebSocket clients connect to the Web App using XMPP over WebSocket which connects with the multiplexor, which in turn, connects with the Instant Messaging Server.

Figure 17-1 XMPP WebSocket Gateway Architecture



Configuring the XMPP WebSocket Gateway

To enable the XMPP WebSocket Gateway:

1. Install and configure Instant Messaging Server.

2. Run the following command to create a ZIP file for the XMPP WebSocket Gateway application that you can move to another machine or keep locally:

```
iwadmin generatezip websocket -c location_of_config_dir/iim.conf.xml -d /tmp/xmpp_websocket.zip
```

where:

- The **-c** option specifies the future location of the configuration file for the WebSocket application.
 - The **-d** option specifies a destination directory and name for the ZIP file.
 - The ZIP file's directory structure and contents are:
 - **/websocket.war**
 - **/lib/imcli.jar**
 - **/lib/imconfutil.jar**
 - **/lib/imcommon.jar**
 - **/sbin/functions**
 - **/sbin/imconfutil**
 - **/config/iim.conf.xml.websocket.template**
 - **/config/websocket_log4j.conf.template**
 - The ZIP file contains:
 - A deployable WAR file (**websocket.war**) for the WebSocket application.
 - The XWS configuration file template (**iim.conf.xml.websocket.template**). The configuration file is the same as the one used by the multiplexor and Instant Messaging Server, and uses the same **imconfutil** command for configuring properties.
 - The log4j template file (**websocket_log4j.conf.template**).
 - The utility for configuring properties in the **iim.conf.xml** file (**imconfutil**).
3. Extract the contents and deploy the **websocket.war** file to WebLogic Server:

```
unzip /tmp/websocket.zip -d /local/websocket_gateway
```
 4. Rename the template files:

```
cd /local/websocket_gateway
cp config/iim.conf.xml.websocket.template InstantMessaging_home/config/iim.conf.xml
cp config/websocket_log4j.conf.template InstantMessaging_home/config/log4j.conf
```
 5. Configure the Instant Messaging Server port to which the multiplexor connects:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop im_mux.serverport=hostname1:port,hostname2:port
```
 6. Modify the configuration properties for logging, monitoring, and so forth:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop property-name=property-value
```

See ["Managing the XMPP WebSocket Gateway"](#) for more information.
 7. Ignore the following **imconfutil** warning:

"Warning: Unable to set file permissions in the dir <path-to-config-dir>, due to Null value for key: iim.user/group"

You can manually modify these file permissions.

8. Start the Instant Messaging Server instance to which the **iim_mux.serverport** is pointing.

```
imadmin start server
```

9. Start Weblogic Server on which the **websocket.war** file is deployed listening on an HTTP port.

```
Weblogic_Instance/bin/startWeblogic.sh
```

The **websocket.log** file (from **log4j.conf**) log the details of the Instant Messaging servers that are connected to this XMPP WebSocket Gateway application. The location of **websocket.log** is configured through **log4j.conf**. The **log4j.conf** template has a default logging of **INFO** level and is turned on by default.

10. Set the initial values for tuning properties for the XMPP WebSocket Gateway.

For more information, see ["Tuning the XMPP WebSocket Gateway"](#).

Managing the XMPP WebSocket Gateway

Managing the XMPP WebSocket Gateway involves:

- [Configuring XMPP WebSocket Gateway Logging](#)
- [Configuring XMPP WebSocket Gateway Monitoring](#)
- [Configuring XMPP WebSocket Gateway Timeout](#)

Configuring XMPP WebSocket Gateway Logging

The XMPP WebSocket Gateway uses log4j for logging purposes. You can change log levels or add new log appenders.

- To enable the DEBUG log level on the XMPP WebSocket Gateway:

```
$configdir/config/log4j.conf .log4j.logger.ws-xmpp=DEBUG, A7
```

- To set a log4j property on the XMPP WebSocket Gateway:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop  
iim.log4j.config=$configdir/config/log4j.conf
```

Configuring XMPP WebSocket Gateway Monitoring

To configure WebSocket Gateway for monitoring:

1. Run the following command to enable monitoring:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_  
mux.monitor.enable = true
```

2. Run the following command to set the common refresh timeout for the multiplexor statistics cache:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_  
mux.monitor.refreshtimeout=60
```

3. Use any JMX client jconsole to connect to the WebLogic Server monitoring port to monitor the gateway.

Configuring XMPP WebSocket Gateway Timeout

To configure a timeout value for the XMPP WebSocket Gateway:

1. Run the following command:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
websocket.session_idletimeout=30
```

2. Run the following command to disable timeout on the XMPP WebSocket Gateway:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
websocket.session_idletimeout=-1
```

Configuring XMPP WebSocket Gateway for Server Pool

You can configure the XMPP WebSocket Gateway for high availability. For more information, see ["About Multiplexor Failover"](#).

Enabling Secure Communication Between the XMPP WebSocket Gateway and Instant Messaging Server

To ensure secure communication between the XMPP WebSocket Gateway and Instant Messaging Server, you must configure the Gateway to use an encrypted channel. Channel refers to a logical connection between client and server. The channel data is encrypted after the XMPP WebSocket Gateway receives data from the client and before sending it to the server.

- To enable secure communication between the XMPP WebSocket Gateway and Instant Messaging Server:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
mux.encrypted_channel=true
```

Secure communication between the XMPP WebSocket Gateway and Instant Messaging Server is enabled by default in the **websocket_log4j.conf.template** file. See ["Configuring the XMPP WebSocket Gateway"](#) for more information.

If the server is configured with trusted certificates then you do not need to configure the Gateway as described in ["Activating TLS on XMPP WebSocket Gateway"](#). See the topic on setting up TLS for Instant Messaging Server in *Instant Messaging Server Security Guide* for configuring the server in TLS mode.

Activating TLS on XMPP WebSocket Gateway

The XMPP WebSocket Gateway provides secure communication by using the TLS protocol. The Gateway uses the Java KeyStore (JKS) to validate certificates from the server. However you can setup your own JKS to trust certificates as described in the "Setting Up TLS for Instant Messaging Server" section in *Instant Messaging Server Security Guide*.

[Table 17-1](#) lists the configuration properties used to enable TLS in XMPP Gateway Server.

Table 17–1 XMPP Gateway Server TLS Configuration Properties

Property	Default Value	Description
iim_mux.sslkeystore	None	Contains the relative path and file name for the server's Java keystore (JKS). For example: <i>InstantMessaging_home/gateway-keystore.jks</i> This property is not required if server is using trusted certificates.
iim_mux.keystorepasswordfile	None	Contains the relative path and the name of the file that contains the password for the keystore. This file should contain the following line: <i>Internal (Software) Token:password</i> where <i>password</i> is the password protecting the keystore. This property is not required if the server is using trusted certificates.
iim_mux.usessl	false	If this value is true , the Gateway uses TLS for channel data encryption.
iim_mux.trust_all_cert	false	If this value is true , the Gateway trusts all certificates, including expired and self-signed certificates, and also adds the certificate information into the log files. If false , the server does not log certificate information and trusts only valid certificates signed by a CA.

Configuring the XMPP/HTTP Gateway

This chapter describes how to configure the HTTPBIND Gateway for Oracle Communications Instant Messaging Server.

About the XMPP/HTTP Gateway

The XMPP/HTTP Gateway provides Instant Messaging access to non-XMPP based clients, such as HTML-based clients and clients behind firewalls that allow HTTP traffic, but does not permit XMPP traffic. The gateway proxies Instant Messaging traffic to the XMPP server on behalf of HTTP clients.

The XMPP/HTTP Gateway is deployed as a web application on the web container.

Instant Messaging Server XMPP/HTTP Gateway Configuration Files

The XMPP/HTTP Gateway uses the following files for configuration:

- Gateway web application configuration file (**web.xml**). The contents of this file determine which gateway configuration file to use. See ["Configuring the XMPP/HTTP Gateway to Use a Non-default Configuration"](#) for information on using a non-default configuration file.
- Gateway configuration file (typically **httpbind.conf**). See ["Configuring the Instant Messaging Server XMPP/HTTP Gateway"](#) for instructions on configuring the gateway. See ["XMPP and HTTP Gateway Configuration Parameters"](#) for a description of **httpbind.conf** file syntax, file location, and a list of configuration parameters in this file.
- Gateway logging configuration file (typically **httpbind_log4j.conf**). See ["Managing Logging for the XMPP/HTTP Gateway"](#) for more information on configuring logging. See ["XMPP/HTTP Gateway log4j Log Configuration File Syntax"](#) for logging configuration file syntax.

Configuring the Instant Messaging Server XMPP/HTTP Gateway

When you run the **configure** utility after installation, you can choose to deploy the XMPP/HTTP Gateway or not. If enabled, the **configure** utility creates a default configuration file (**httpbind.conf**) for the gateway. You can change the configuration by modifying the values in this file. See ["XMPP and HTTP Gateway Configuration Parameters"](#) for a description of **httpbind.conf** file syntax, file location, and a list of configuration parameters in this file, or refer to the instructions in this section.

In addition, when you choose to deploy the gateway during initial configuration, the **configure** utility creates a **.war** file in the *InstantMessaging_home/work* directory and

then deploys this file on GlassFish Server in the directory you specified for the code base.

You can also configure the gateway to use a non-default configuration file by modifying the values in the **web.xml** file, which is deployed with the client resources on the GlassFish Server.

The instructions in this section assume the gateway configuration file is **httpbind.conf**. If you are using a non-default configuration file, substitute your configuration file for **httpbind.conf** in the instructions.

Any time you make a change to **httpbind.conf**, you will need to restart the XMPP/HTTP Gateway.

See ["Managing Logging for the XMPP/HTTP Gateway"](#) for instructions on configuring logging for the gateway.

Enabling or Disabling the Instant Messaging Server XMPP/HTTP Gateway

You enable the gateway by running the **configure** utility. You can disable the gateway later by using GlassFish Server tools.

To enable or disable the XMPP/HTTP gateway:

1. To enable the gateway:
 - a. Invoke the **configure** utility.

For more information, see the topic on configuring Instant Messaging Server after installation in *Instant Messaging Server Installation and Configuration Guide*.
 - b. Choose to deploy the gateway when prompted.
2. To disable the gateway, use GlassFish Server tools to disable the web application.

Manually Configuring HTTPBIND

The high-level steps to manually configure HTTPBIND include:

1. Configuring HTTPBIND
2. Configuring GlassFish Server
3. Configuring Instant Messaging Server
4. Deploying HTTPBIND

To manually configure HTTPBIND:

1. Run the **iwadmin** command to generate the zip file for HTTPBIND.

```
iwadmin generatezip httpbind -c /opt/sun/comms/im/config/httpbind.conf -d /tmp/httpbind.zip
```

See ["iwadmin Command Reference"](#) for more information.

2. Unzip the generated zip file.

```
unzip /tmp/httpbind.zip -d /tmp/
```

3. Copy the **config** template file from the zipped folders's **config** directory to the **config** directory of httpbind. In the following example, **/opt/sun/comms/im/config** is the **config** directory of httpbind.

```
cp /tmp/httpbind/config/httpbind.conf.template /opt/sun/comms/im/config/httpbind.conf
```

4. Copy the log template file of httpbind to the **config** directory

```
cp /tmp/httpbind/config/httpbind_log4j.conf.template
/opt/sun/comms/im/config/httpbind_log4j.conf
```

See "[Managing Instant Messaging Server Logging by Using log4j](#)" for log-related configurations.

5. Edit the **httpbind.conf** file (by default, located in the **/opt/sun/comms/im/config** directory) to set the following properties:

- Domain: **default.domains=abc.com**
- Multihosting to **false**: **default.multihosting=false**
- Host name and port of Instant Messaging Server host:
default.hosts=example.com:5269
- Component jid of httpbind: **default.componentjid=httpbind.abc.com**
- Password: **default.password=password**

To set the password in plain text, comment out the following parameters:

```
#httpbind.component.password.cipher.delegate=DELEGATE_CLASS
#httpbind.component.password.cipher=CIPHER_CLASS
```

See "[Using Encrypted Passwords](#)" to use encrypted passwords.

6. Set the path to the **log4j** file used by HTTPBIND:

```
httpbind.log4j.config=/opt/sun/comms/im/config/httpbind_log4j.conf
```

See the following sections for additional HTTPBIND-related configuration settings.

7. To configure GlassFish Server for HTTPBIND, on the GlassFish Server, enable Comet support:

```
asadmin set --user admin --secure=true --host localhost --port 4848
'server.http-service.http-listener.http-listener-1.property.cometSupport=true'
asadmin set --user admin --secure=true --host localhost --port 4848
'server.http-service.http-listener.http-listener-2.property.cometSupport=true'
```

8. Run the following commands to configure Instant Messaging Server:

```
imconfutil -u set-listener-prop -c InstantMessaging_home/config/iim.conf.xml
s2s protocols=s2s,component,c2s
imconfutil -u set-prop -c InstantMessaging_home/config/iim.conf.xml iim_
server.useport=true
imconfutil add-component -c InstantMessaging_home/config/iim.conf.xml
id=httpbind1 jid=httpbind.abc.com password=password
```

Ensure that the jid and password match with the ones that you previously configured in the **httpbind.conf** file.

9. To deploy HTTPBIND, deploy the **httpbind.war** file, from the unzipped **httpbind.zip** file previously generated, to Glassfish Server.

10. Restart GlassFish Server.

11. Restart Instant Messaging Server.

When you manually configure HTTPBIND, ensure that the files in the **config** directory, and encryption key files if you are using password encryption, have read

permissions set for GlassFish Server. When you run the **configure** utility, the utility ensures that the permissions are correct when you configure HTTPBIND.

Configuring Concurrent Requests Handled by the XMPP/HTTP Gateway

Ensure that you are familiar with the JEP 124 draft standard.

To configure concurrent requests handled by the XMPP/HTTP gateway:

1. Open **httpbind.conf**.
2. Set the **httpbind.requests** parameter to the maximum number of concurrent requests a single client can send to the gateway. The default is **2**. For example:

```
httpbind.requests=2
```

If the value of this parameter is less than the value for the JEP 124 **hold** attribute in the client request, the value for this parameter will be set to **hold+1**. Do not set this parameter to **1**, as doing so could severely degrade performance. See ["Setting the JEP 124 hold Attribute for Client Requests to the XMPP/HTTP Gateway"](#) and [Table 34-1, "XMPP/HTTP Gateway Configuration Parameters in httpbind.conf"](#) for more information on the **httpbind.hold** parameter.

3. Save and close **httpbind.conf**.
4. Restart the gateway by using the tools provided by GlassFish Server.

Setting the JEP 124 hold Attribute for Client Requests to the XMPP/HTTP Gateway

Ensure that you are familiar with the JEP 124 draft standard.

To set the JEP 124 hold attribute for client requests to the XMPP/HTTP gateway:

1. Open **httpbind.conf**.
2. Set the **httpbind.hold** parameter to the maximum value you want the gateway to allow for the **hold** attribute in the client request. The default is **5**. For example:

```
httpbind.hold=5
```

If the **hold** value sent by the client is greater than the gateway's **hold** value, the gateway's **hold** value is used.

3. Save and close **httpbind.conf**.
4. Restart the gateway by using the tools provided by GlassFish Server.

Specifying the Allowed Client Inactivity Time for the XMPP/HTTP Gateway

To specify allowed client inactivity time for the XMPP/HTTP gateway:

1. Open **httpbind.conf**.
2. Set the **httpbind.inactivity** parameter to the time in seconds after which you want the gateway to terminate idle connections. The default is **180** seconds. For example:

```
httpbind.inactivity=180
```

If clients do not poll the gateway before this time elapses, the gateway terminates the connection.

3. Save and close **httpbind.conf**.

4. Restart the gateway by using the tools provided by GlassFish Server.

Setting the content-type HTTP Header for the XMPP/HTTP Gateway

To set the content-type HTTP header for the XMPP/HTTP gateway:

1. Open **httpbind.conf**.
2. Set the **httpbind.content_type** parameter to the content-type you want the gateway to use if the client does not specify one in its initial request. The default is **text/xml; charset=utf-8**. For example:

```
httpbind.content_type=text/xml; charset=utf-8
```

3. Save and close **httpbind.conf**.
4. Restart the gateway by using the tools provided by GlassFish Server.

Setting the Round Trip Delay for the XMPP/HTTP Gateway

The round trip delay is the amount of time, in seconds, you want to allow in addition to time-outs for round trips between gateway and client. This helps to account for network latencies.

To set the round trip delay for the XMPP/HTTP gateway:

1. Open **httpbind.conf**.
2. Set the **httpbind.round_trip_delay** parameter as required. Setting this value too high might degrade performance. The value is in seconds. The default is one second. For example:

```
httpbind.round_trip_delay=1
```

Setting this value too high may degrade performance. Consider the general latency in your network before changing this parameter.

3. Save and close **httpbind.conf**.
4. Restart the gateway by using the tools provided by the GlassFish Server.

Setting the XMPP/HTTP Gateway Default Response Time

To set the XMPP/HTTP gateway default response time:

1. Open **httpbind.conf**.
2. Set the **httpbind.wait_time** parameter as required.

The client is guaranteed a response from the XMPP/HTTP Gateway within the wait time you designate with this parameter. Consider the speed of your network when setting this parameter.

Do not set the value so low that the XMPP/HTTP Gateway is unlikely to be able to send the request in time.

The value is in seconds. The default is 120 seconds. For example:

```
httpbind.wait_time=120
```

If the value set for the client is greater than the value for the gateway, the gateway wait time is used.

3. Save and close **httpbind.conf**.

4. Restart the gateway by using the tools provided by GlassFish Server.

Configuring an XMPP/HTTP Gateway in an Instant Messaging Server Gateway Pool

To configure an XMPP/HTTP gateway in a gateway pool:

1. Open **httpbind.conf**.
2. To configure the gateway as part of a deployment with an Instant Messaging Server gateway pool:
 - a. Set the **httpbind.pool.support** parameter to **true**:

```
httpbind.pool.support=true
```
 - b. Set the **httpbind.pool.nodeId** parameter to the full URL of the gateway.

The URL is used as the gateway's **nodeId**. The **nodeId** must be unique within the server pool. The gateway uses the **nodeId** to determine whether it must service a received request or forward the request to another gateway in the pool.
3. To configure the gateway not to work within a gateway pool, set the **httpbind.pool.support** parameter as follows:

```
httpbind.pool.support=false
```
4. Save and close **httpbind.conf**.
5. Restart the gateway by using the tools provided by GlassFish Server.

Configuring the List of Key IDs for Supported XMPP/HTTP Gateway Domains

To configure the list of key IDs for supported XMPP/HTTP gateway domains:

1. Open **httpbind.conf**.
2. Set the **httpbind.config** parameter to the list of IDs you want the gateway to use.

For each domain you must specify a separate ID for this parameter. For example:

```
httpbind.config=gwdomain-id
```

Where *gwdomain-id* is the identifier you want to use for the domain.

For example:

```
httpbind.config=siroe.com
```
3. For each *gwdomain-id* you specify, add the following parameters to the **httpbind.conf** file:

```
gwdomain-id.domain=domain-name
gwdomain-id.hosts=gateway-host
gwdomain-id.componentjid=component-jid
gwdomain-id.password=password
```

where:

- *gwdomain-id* is the ID specified for the gateway in **httpbind.config** in the previous step.
- *domain-name* is the domain in which the identified gateway runs.

- *gateway-host* is a comma-separated or space-separated list of the fully-qualified domain name (FQDN) and port number of the gateway hosts that support this domain.
- *component-jid* is the component JID of the gateway.
- *password* is the password of the identified gateway.

For example, if *gwdomain-id* is set to **siroe**:

```
siroe.domain=siroe.com
siroe.hosts=gateway.siroe.com:5222
siroe.componentjid=http.gateway.siroe.com
siroe.password=gatewaypassword
```

See "[Gateway Domain ID Key Parameters for httpbind.config](#)" for more information about these key parameters.

4. Save and close **httpbind.conf**.
5. Restart the gateway by using the tools provided by GlassFish Server.

Configuring the XMPP/HTTP Gateway to Use a Non-default Configuration

To configure a non-default configuration for the XMPP/HTTP gateway:

1. On the web container, edit **web.xml**.
Use your web container's tools to edit this file.
2. Change the value for the **httpbind.conf** file parameter to the location of the configuration file you want the gateway to use.

Using Encrypted Passwords

You can assign encrypted passwords to XMPP/HTTP gateway domains listed in the **httpbind.conf** file.

To assign an encrypted password to a gateway:

1. Open the **httpbind.conf** file and set the **httpbind.component.password** properties:

```
httpbind.component.password.cipher.delegate=com.sun.im.tools.passwordtool.Crypto
httpbind.component.password.cipher=com.sun.im.tools.passwordtool.CommsClientCipher
```

2. From the directory containing the **httpbind.conf** file, generate a password key and password using the Instant Messaging Server **passwordtool** command.

See "[passwordtool Command Reference](#)" for more information.

For example, the following commands generate an encrypted password from the clear text password *abcd*:

```
cd httpbind_config_dir
passwordtool httpbind generate-key
passwordtool httpbind generate abcd
MmHRfLCIB0ej5KGDqLC45Q==
```

3. In the **httpbind.conf** file, set the **gwdomain-id.password** property to the encrypted password, as in the following example.

The gateway ID is **siroe**.

```
siroe.password=MmHRfLCIB0ej5KGDqLC45Q==
```

Adding a New Hosted Domain Without Restarting GlassFish Server

The **gwdomain-id.multihosting** parameter in the **httpbind.conf** file, if set to **true**, allows a packet destined to a domain, which is not preconfigured in **httpbind.conf**, to be sent to Instant Messaging Server. You use this parameter for a hosted domain setup. The default value for this parameter is **true**.

Using StartTLS to Secure Communication Between XMPP/HTTP and IM Server

The XMPP/HTTP Gateway only supports **StartTLS** for secure communications. The gateway always attempts to use **StartTLS** if it is available. See *Instant Messaging Server Security Guide* for more information.

Managing Logging for the XMPP/HTTP Gateway

This section describes configuring the level of logging for the XMPP/HTTP Gateway, enabling or disabling logging entirely, and changing the location of the gateway log file or the gateway log configuration file.

For more information about the log4j format supported by Instant Messaging Server, see the Apache Logging Services website, at:

<http://logging.apache.org>

Enabling or Disabling Logging for the XMPP/HTTP Gateway

You can enable or disable logging for the gateway in the following ways:

- Adding or removing the value for the **httpbind.log4j.config** parameter in **httpbind.conf**.
- (Recommended) Modifying the configuration within the gateway's **log4j** configuration file (**httpbind_log4j.conf**).

Under most circumstances, you should modify the configuration in the **httpbind_log4j.conf** file itself, leaving the **httpbind.log4j.config** parameter set to the location of the **httpbind_log4j.conf** file. This procedure describes modifying the configuration within the **httpbind_log4j.conf** file.

To enable or disable logging for the XMPP/HTTP gateway:

1. Open the **httpbind_log4j.conf** file.

This file is stored at the location you specified in **httpbind.conf** file as the value for the **httpbind.log4j.config** parameter. By default the file is stored in the following directory under the default Instant Messaging Server instance:

```
InstantMessaging_cfg/httpbind_log4j.conf
```

2. To disable logging for the gateway, set the **log4j.logger.httpbind** parameter as follows:

```
log4j.logger.httpbind=OFF
```

3. To enable logging, set the **log4j.logger.httpbind** parameter to the desired logging level.

For example:

```
log4j.logger.httpbind=ERROR
```

See [Table 14–1, "Logging Levels for Instant Messaging Server Components"](#) for a list of valid logging levels you can use.

4. Save and close **httpbind_log4j.conf**.

Changing the Location of the XMPP/HTTP Gateway Log Configuration File

To change the location of the XMPP/HTTP gateway log configuration file:

1. Open **httpbind.conf**.
2. Set the value of the **httpbind.log4j.config** parameter to the location of the XMPP/HTTP Gateway log configuration file.
3. Save and close **httpbind.conf**.
4. Restart the gateway by using the tools provided by GlassFish Server.

Setting the XMPP/HTTP Gateway Log File Location on Linux

On Linux systems, after you install and configure the XMPP/HTTP Gateway, you must modify the location of the default log file for the XMPP/HTTP gateway in **httpbind_log4j.conf**.

To set the XMPP/HTTP gateway log file location on Linux:

1. Open the **httpbind_log4j.conf** file.

This file is stored at the location you specified in **httpbind.conf** file as the value for the **httpbind.log4j.config** parameter. By default the file is stored in the following directory under the default Instant Messaging Server instance:

InstantMessaging_cfg/httpbind_log4j.conf

2. Set the value of the **log4j.appender.appender_ID.file** parameter to the location where log files are stored.

Changing the Location of the XMPP/HTTP Gateway Log File

Ensure that you are familiar with the log4j syntax and general implementation described at Apache Logging Services, at:

<http://logging.apache.org>

To change the location of the XMPP/HTTP gateway log file:

1. Open **httpbind_log4j.conf**.

This file is stored at the location you specified in **httpbind.conf** file as the value for the **httpbind.log4j.config** parameter. By default the file is stored in the following directory under the default Instant Messaging Server instance:

InstantMessaging_cfg/httpbind_log4j.conf

2. Set the value for the **log4j.appender.appender-ID** parameter to the location where you want to store the log file.
3. Save and close **httpbind_log4j.conf**.
4. Restart the web container.

Using a Non-default Log File Location for the XMPP/HTTP Gateway

If you choose to use a location for logs other than the default, you must modify the location of the default log file for the XMPP/HTTP gateway in **httpbind_log4j.conf**.

To use a non-default log file location for the XMPP/HTTP gateway:

1. Open the **httpbind_log4j.conf** file.

This file is stored at the location you specified in **httpbind.conf** file as the value for the **httpbind.log4j.config** parameter. By default the file is stored in the following directory under the default Instant Messaging Server instance:

InstantMessaging_cfg/httpbind_log4j.conf

2. Set the value of the **log4j.appender.appender_ID.file** parameter to the location where log files are stored.

Setting the XMPP/HTTP Gateway Logging Level

Ensure that you are familiar with the log4j syntax and general implementation described at Apache Logging Services, at:

<http://logging.apache.org>

To set the XMPP/HTTP gateway logging level:

1. Open **httpbind_log4j.conf**.

This file is stored at the location you specified in **httpbind.conf** file as the value for the **httpbind.log4j.config** parameter. By default the file is stored in the following directory under the default Instant Messaging Server instance:

InstantMessaging_cfg/httpbind_log4j.conf

2. Set the **log4j.logger.httpbind** parameter to the desired logging level.

For example:

```
log4j.logger.httpbind=ERROR
```

See [Table 14–1, "Logging Levels for Instant Messaging Server Components"](#) for a list of valid logging levels you can use.

XMPP/HTTP Gateway log4j Log Configuration File Syntax

For more information about the log4j syntax and general implementation, see Apache Logging Services at:

<http://logging.apache.org>

The XMPP/HTTP Gateway log configuration file syntax is as follows.

```
log4j.logger.httpbind=logging-level,Appender-ID
# DEFAULT TO RollingFileAppender
log4j.appender.Appender-ID=org.apache.log4j.RollingFileAppender
log4j.appender.Appender-ID.file=log-dir/httpbind.log
log4j.appender.Appender-ID.append=true|false
log4j.appender.Appender-ID.maxBackupIndex=7
log4j.appender.Appender-ID.maxFileSize=max-log-file-size
log4j.appender.Appender-ID.layout=org.apache.log4j.PatternLayout
log4j.appender.Appender-ID.layout.ConversionPattern=log-entry-syntax
```

XMPP/HTTP Gateway Log Configuration File Example

The following is an example of an XMPP/HTTP Gateway log configuration file:

```
log4j.logger.httpbind=ERROR, A7
# DEFAULT TO RollingFileAppender
log4j.appender.A7=org.apache.log4j.RollingFileAppender
# log4j.appender.A7.file=$(logdir)/httpbind.log
log4j.appender.A7.file=_log-dir_/httpbind.log
log4j.appender.A7.append=true
log4j.appender.A7.maxBackupIndex=7
log4j.appender.A7.maxFileSize=5mb
log4j.appender.A7.layout=org.apache.log4j.PatternLayout
log4j.appender.A7.layout.ConversionPattern=[%d{DATE}] %-5p %c [%t] %m%n
```

Configuring External Service Discovery

This chapter describes the External Service Discovery feature and how to configure it for Oracle Communications Instant Messaging Server.

About External Service Discovery

External Service Discovery enables the XMPP server to support and share information about services external to the XMPP server, mainly TURN (Traversal Using Relay NAT) and STUN (Session Traversal Utilities for NAT) server details, with clients.

This feature is available as a plugin that handles different types of packets, for example, IQ, and Presence and Message. Currently, Instant Message Server supports only the custom IQ handler.

Configuring External Discovery

To configure the XMPP/HTTP Gateway to enable the External Service Discovery Plugin:

1. Change to the `/opt/sun/comms/im/sbin` directory.

```
cd /opt/sun/comms/im/sbin
```

2. Run the following command to enable the External Service Discovery plugin:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml add-plugin id=XEP0215  
type=iq  
enable=true handlerclass=oracle.communications.ucs.ocim.handler.  
ExtDiscoHandler  
namespace=urn:xmpp:extdisco:2
```

3. Run the following command to add the provider class for External Service Discovery:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_  
server.ext_  
disco.provider=oracle.communications.ucs.ocim.provider.sample.FileServiceDiscoP  
rovider
```

Note: Instead of using the sample file-based provider, you can specify a differently implemented provider class, such as a provider that fetches STUN and TURN server details from a database.

- Point the file-based provider to fetch STUN and TURN data from a CSV file by creating a file named **extServiceDisco.csv** in the **config** directory:

```
vi ../config/extServiceDisco.csv
```

- Add STUN and TURN server details in the following format:

```
# expires, host, name, password, port, restricted, transport, type, username
```

For example:

```
1971-07-21T02:56:15Z, stun.example.com, "STUN Server", jj929jkj5sadjfj93v3n,
9998, false, udp, stun, nb789321kjlskjfdb7g
```

Table 19–1 describes the attributes of services external to the XMPP server, their definition, and whether their inclusion is required, recommended, or optional.

Table 19–1 Attributes of Services External to the XMPP Server

Attribute	Definition	Inclusion
expires	A timestamp indicating when the provided user name and password credentials expire. The format must adhere to the dateTime format specified in XMPP Date and Time Profiles standard (XEP-0082), and must be expressed in UTC. For more information, see the XEP specification at: http://xmpp.org/extensions/xep-0082.html	Optional
host	Either a fully qualified domain name (FQDN) or an IP address (IPv4 or IPv6).	Required
name	A friendly (human-readable) name or label for the service.	Optional
password	A service- or server-generated password for use at the service.	Optional
port	The communications port used by the host.	Recommended
restricted	A boolean value indicating that user name and password credentials are required and must be requested if not already provided. For more information, see the discussion about requesting credentials at: http://xmpp.org/extensions/xep-0215.html#credentials	Optional
transport	The underlying transport protocol used when communicating with the service (typically either TCP or UDP).	Recommended
type	The service type as registered with the XMPP Registrar. For more information, see: http://www.xmpp.org/extensions/xep-0053.html	Required
username	A service- or server-generated user name for use by the service.	Optional

- If not already present, create a **sysServiceDisco.acl** file to specify which users are allowed access to fetch TURN and STUN details:

```
vi ../config/acls/sysServiceDisco.acl
```

- Add the JID of users authorized to receive the services information. For example, if all the users are to be allowed, add the following line to the **acl** file:

```
d:true
```

If only two users having JIDs such as **bard1@example.com** and **bard2@example.com** are to be allowed, add the following lines to the **acl** file:

```
u:bard1@example.com  
u:bard2@example.com  
d:false
```

Configuring the SIP Gateway

This chapter describes the Oracle Communications Instant Messaging Server SIP/SIMPLE gateway.

About the SIP Gateway

Instant Messaging Server implements a SIP/SIMPLE (Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions/Session Initiation Protocol) gateway. The gateway enables federation (inter-domain) and translation between the two protocols, and interoperation between XMPP and SIP/SIMPLE servers (for example, OpenSER).

The Instant Messaging Server SIP gateway enables:

- Users to subscribe to the presence of contacts on an external SIP/SIMPLE network
- Messaging between users on the Instant Messaging server and their contacts on an external SIP/SIMPLE network
- Enforces the privacy preferences of the XMPP Instant Messaging Server users

Enabling of SIP/SIMPLE federation is optional. You can choose a particular set of SIP domains for which federation is allowed or you can enable open federation. If both XMPP and SIP open federation are enabled, and a domain has both XMPP and SIP servers, XMPP federation is the preferred option. You can also choose to blacklist certain domains from open federation. See ["Configuring Federated Communication Between Instant Messaging Servers"](#) and ["Configuring the SIP Gateway"](#) for more information.

Currently, the Instant Messaging Server SIP gateway does not support:

- File transfer between SIP and XMPP clients (users)
- Group chat (text conferencing)
- Audio/video chat

Instant Messaging Server can also provide server-to-server federation support between any standard XMPP server and the SIP/SIMPLE Federation Service Gateway. The Instant Messaging Server SIP Gateway uses either Jabber Component Protocol or Server-to-Server federation for interoperability. The Jabber Component Protocol is only supported with Instant Messaging server. For any other third-party XMPP server, you must configure the Server to Server federation option.

You can enable server-to-server federation between any standard XMPP server and the SIP Gateway in the following configurations:

- Plain Text and Dialback, also known as Verified Federation

A server accepts a connection from a peer only after the identity of the peer has been weakly verified through Dialback, based on information obtained from the Domain Name System (DNS) and verification keys exchanged in-band over XMPP. However, the connection is not encrypted. The use of identity verification effectively prevents domain spoofing, but federation requires proper DNS setup and is still subject to DNS poisoning attacks.

- TLS and Dialback, if self signed certificates are provided, also known as Encrypted Federation

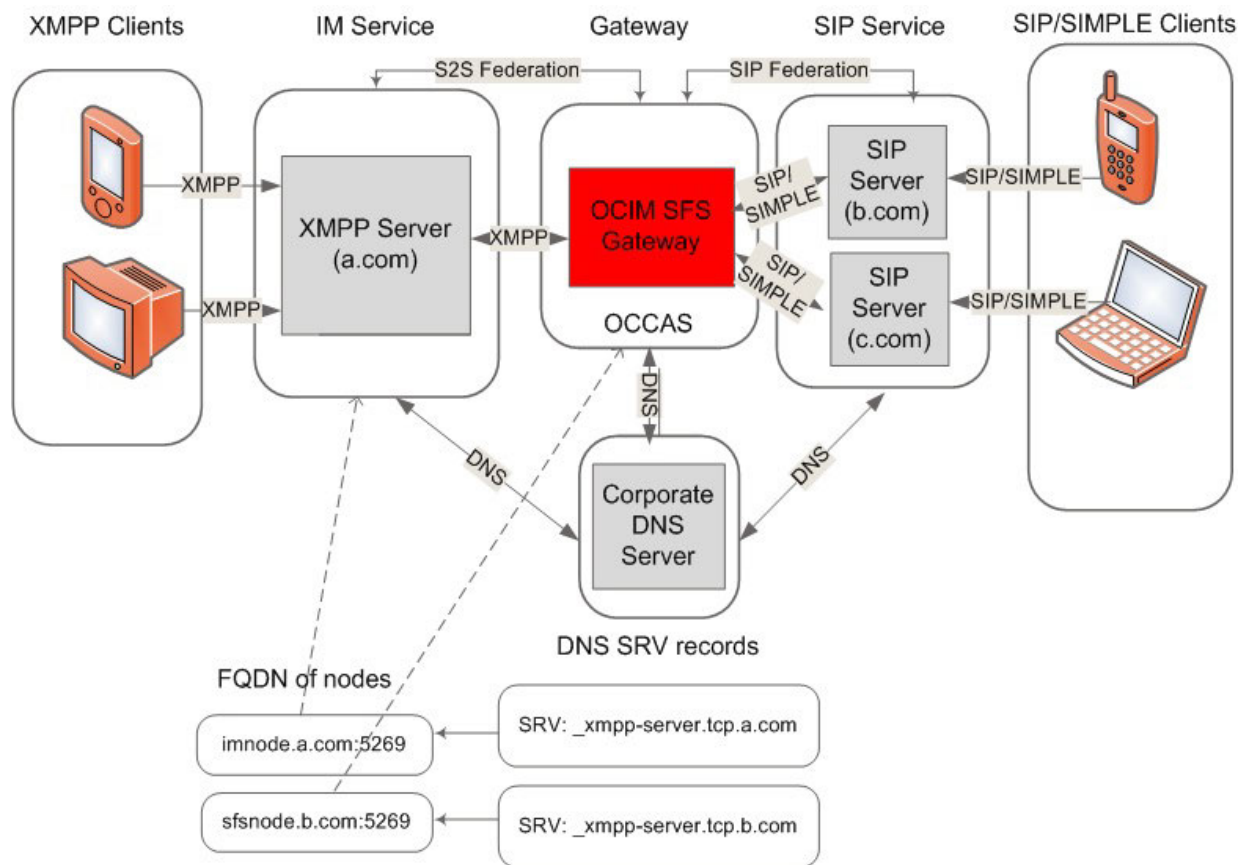
A server accepts a connection from a peer only if the peer supports Transport Layer Security (TLS) and the peer presents a digital certificate. However, the certificate may be self-signed, in which case mutual authentication is typically not possible. Therefore, after STARTTLS negotiation the parties proceed to weakly verify identity using Dialback. This combination results in an encrypted connection with weak identity verification.

- TLS, also known as Trusted Federation

A server accepts a connection from a peer only if the peer supports TLS and the peer presents a digital certificate issued by a trusted root certification authority (CA). The list of trusted root CAs is determined by local service policy, as is the level of trust accorded to various types of certificates (for example, Class 1, Class 2, or Class 3). The use of trusted domain certificates effectively prevents DNS poisoning attacks but makes federation more difficult as typically such certificates are not easy to obtain.

SIP Gateway Architecture

[Figure 20–1](#) shows the SIP gateway architecture.

Figure 20–1 SIP Gateway Architecture

This figure shows that the SIP Federation Service is implemented as a SIP servlet, deployed within the Oracle Communications Converged Application Server. XMPP server users are able to exchange presence and chats with SIP users over the XMPP protocol (by way of the SIP gateway), while SIP users do the same over the SIP/SIMPLE protocol through the SIP/SIMPLE server. The SIP gateway uses either Jabber Component Protocol or Server-to-Server Federation to communicate with the XMPP server. For Server-to-Server federation, you must configure DNS SRV records for both XMPP and SIP domains. The SIP gateway converts SIMPLE requests to the appropriate XMPP format and sends them to the XMPP server over either component or federated connection. SIMPLE requests are acknowledged and responded to appropriately by using the SIP servlet API. Similarly, the SIP gateway converts XMPP requests or responses received from the XMPP server to the appropriate SIMPLE requests and then sends them to the SIMPLE clients. The SIP gateway maintains both SIMPLE and XMPP user subscription states. The SIP gateway needs to interact with the XMPP server to authorize presence subscriptions and obtain SIP user presence notifications.

Configuring the SIP Gateway

This section explains how to configure the SIP gateway.

Prerequisites for Configuring the SIP Gateway

You need the following components to configure and use the SIP Gateway:

- Instant Messaging Server or any standard XMPP instant messaging server

- Oracle Communications Converged Application Server 5.1
- XMPP capable client, such as Pidgin
- SIP/SIMPLE capable client, such as Jitsi

XMPP and SIP domains federating with each other must be resolvable to the respective XMPP and SIP hosts through DNS Service records (SRV records) for communication to work between them.

When using the SIP Gateway in federation mode, you must ensure that the XMPP domain SRV record points to the SIP Gateway host for the SIP network, and that the SIP domain SRV record points to the SIP Gateway host for XMPP network.

Configuring the SIP Gateway in Component Mode

Perform the following steps on the Instant Messaging Server host:

1. Install and configure Instant Messaging Server. For information, see *Instant Messaging Server Installation and Configuration Guide*.

2. Enable open federation by running the following command:

```
imconfutil set-prop -c InstantMessaging_home/config/iim.conf.xml iim_
server.federation.policy=OPEN
```

3. Create a SIP WAR file by running the following command:

```
/opt/sun/comms/im/sbin/create_sip_war -c component -h coms-121.example.com -p
5269 -j sip.coms-121.example.com -t true -o imfed.war
```

4. When prompted, enter the password.

5. Add the SIP component by running the following command:

```
imconfutil add-component -c InstantMessaging_home/config/iim.conf.xml id=fed
jid=sip.coms-121.example.com password=password
```

Configuring SIP Gateway in Federation Mode

You can configure server-to-server federation support between any standard XMPP server and a SIP/SIMPLE Federation Service Gateway.

Before you can configure Instant Messaging Server for the SIP Gateway using TLS, you must first create an SSL certificate. For more information, see the topic on setting up TLS in *Instant Messaging Server Security Guide*.

Note: The Instant Messaging Server SIP Gateway supports XMPP federation with single XMPP and SIP domains.

Enabling S2S Communication Using TLS and SASL-External

To enable S2S communication using TLS and SASL-external, use the **create_sip_war** command to create the WAR file:

```
create_sip_war -c s2s -h xmppserver.domainname -j sipserver.domainname -t true -k
/tmp/trust.jks -f /tmp/sslpassword.conf -o imfed.war
```

In this command:

-k specifies the path to the keystore file.

-f specifies the path to the file that contains the password for the keystore.

Enabling S2S Communication Using TLS and Dialback

To enable S2S communication using TLS and dialback:

1. Create a default configuration file, if it does not already exist.

```
create_sip_war -c s2s -h xmppserver.domainname -j sipserver.domainname -t true
-k /tmp/trust.jks -f /tmp/sslpassword.conf -o imfed.war
```

2. If you are using a self-signed certificate for the SIP Gateway, set the **iim_server.trust_all_cert** option to **true**.

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop iim_
server.trust_all_cert='true'
```

3. Regenerate the WAR file.

```
create_sip_war -c s2s -h xmppserver.domainname -j sipserver.domainname -t true
-k /tmp/trust.jks -f /tmp/sslpassword.conf -o imfed.war
```

4. When prompted, enter the dialback secret.

Enabling S2S Communication Using Plain Text and Dialback

To enable S2S communication using plain text and dialback:

1. Use the **create_sip_war** command to create the WAR file:

```
create_sip_war -c s2s -h xmppserver.domainname -j sipserver.domainname -t false
-o imfed.war
```

2. When prompted, enter the dialback secret.

Configuring Logging for the SIP Gateway

To configure logging for the SIP gateway:

1. To change the log level or log location of the **imfed.log** file, edit the **/opt/sun/comms/im/lib/log4j-sip.conf** file.
2. Make the necessary changes.
3. Redeploy the WAR file.

Configuring the Oracle Communications Converged Application Server

Perform the following steps on the Oracle Communications Converged Application Server host:

1. Set up Oracle Communications Converged Application Server in basic domain mode. For more information, see *Converged Application Server Documentation* at: http://docs.oracle.com/cd/E17645_01/index.htm
2. Deploy the SIP WAR file generated in the preceding procedure, "Enabling S2S Communication Using Plain Text and Dialback".
3. Restart XMPP server.

If you are using Instant Messaging Server for federation, run the following command:

```
imadmin start
```

4. Restart the Oracle Communications Converged Application Server. See *Starting and Stopping Servers*:
http://docs.oracle.com/cd/E17645_01/doc.50/e17647/opg_starting.htm#CHDHDCFB
5. After restarting the Oracle Communications Converged Application Server, you must enable DNS lookup in the SipServer General setting page.

Testing the SIP Gateway

To test the SIP gateway:

1. On the SIP host, ensure that you have configured XML Configuration Access Protocol (XCAP) and created users.
2. On the Instant Messaging server, log in an XMPP user.
3. On the SIP host, log in a SIP user by using SIP Communicator.
4. Verify that the SIP user is able to add the XMPP user and vice-versa. The two users should be able to chat with each other and see each other's presence.

Troubleshooting the SIP Gateway

When troubleshooting the SIP gateway, make use of the SIP Federator's log file, **imfed.log**. You set the location of the **imfed.log** file before generating the WAR file. By default, the file resides in the **/tmp** directory.

Configuring DNS for XMPP and SIP Federation

XMPP and SIP domains federating with each other must be resolvable to the respective XMPP and SIP hosts through DNS Service records (SRV records) for communication to work between them. See "[Configuring DNS for XMPP Federation](#)" and "[Configuring DNS for SIP Federation](#)" for more information.

Configuring the SMS Gateway

This chapter describes the Short Message Service (SMS) gateway feature and how to configure it for Oracle Communications Instant Messaging Server.

About the SMS Gateway

The SMS gateway feature enables the Instant Messaging server to deliver chat messages and alerts in the form of SMS to Instant Messaging users who are offline. This feature provides a streamlined instant messaging experience by forwarding messages to users' mobile phones when they are offline. The SMS gateway uses the SMPP (short message peer-to-peer) protocol and XMPP (Extensible Messaging and Presence Protocol) for messaging services.

The following list describes the SMS gateway terms:

- **SMS:** Short Message Service is a wireless messaging service that permits the transmission of a short text message from and to a digital wireless terminal.
- **SMSC:** Short Message Service Center is a network element in the mobile telephone network that delivers SMS messages to mobile devices.
- **SMPP:** Short Message Peer-to-Peer protocol is a telecommunication protocol used for exchanging SMS messages between SMS entities. For example, short message service centers.
- **XMPP:** Extensible Messaging and Presence Protocol is an open Extensible Markup Language (XML) protocol for near-real-time messaging, presence, and request-response services.

Configuring the SMS Gateway

To enable the SMS gateway feature, you must configure the Instant Messaging server and client.

SMS and Server Configuration Properties

[Table 21-1](#) shows the SMS configuration properties.

Table 21–1 SMS Gateway Properties

Property	Default Value	Description
<code>lsmgw.imadmin.enable</code>	false	Enables or disables the SMS gateway. If set to true , you can start the SMS gateway by using the imadmin command.
<code>smsgw.jid</code>	None	A jabber ID (JID) to bind the SMS gateway to the Instant Messaging server. The value should be the same as the value that you define for the smppbind.jid property.
<code>smsgw.password</code>	None	Password to authenticate the SMS gateway to the Instant Messaging server. The value should be the same as the value that you define for the smppbind.password property.
<code>smsgw.iim_server</code>	None	Host name and port number of the Instant Messaging server.
<code>smsgw.sms_limit</code>	-1	Number of messages that can be sent per hour. The default value is -1 and it indicates that unlimited number of SMS messages that can be sent per hour.
<code>smsgw.sms_queue_capacity</code>	512	Maximum number of messages that can be queued for SMS delivery.
<code>smsgw.im_char_limit</code>	500	Maximum number of characters that you can specify in one message. If the number of characters is greater than the specified value, the message is rejected.
<code>smpp.smsc_ip_address</code>	None	IP address or host name of the SMSC.
<code>smpp.smsc_port</code>	2775	Port number of the SMSC.
<code>smpp.bind_id</code>	None	Identifier used to bind the SMS gateway to the SMSC.
<code>smpp.bind_password</code>	None	Password to authenticate the SMS gateway to the SMSC.
<code>smpp.sender_id</code>	None	Sender ID of the outgoing SMS.

Table 21–2 shows the Instant Messaging server properties that you use to enable SMS.

Table 21–2 Instant Messaging Server Properties to Enable SMS

Property	Default Value	Description
<code>iim_server.components</code>	None	List of component identifiers that should have smppbind . For example, httpbind , smppbind .
<code>iim_agent.smppbind.enable</code>	smppbind.password	Enables the Instant Messaging server to identify the SMS gateway.
<code>smppbind.jid</code>	None	A jabber JID for binding the SMS gateway to the Instant Messaging server.
<code>smppbind.password</code>	None	Password to authenticate the SMS gateway to the Instant Messaging server.

Server-Side Configuration

You can configure the SMS gateway feature by either running the **configure** utility or the **imconfutil** command.

Configuring the SMS Gateway by Using the imconfutil Command

To configure the SMS gateway by using the **imconfutil** command:

1. Ensure that Instant Messaging Server has been installed.
2. Add the SMS gateway component by using the **add-component** command.

3. Set the SMS gateway properties by using the **set-prop** command

Example: Configuring the SMS Gateway on the Same Host as the Instant Messaging Server

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml add-component
id=msggateway jid=smpplib.example.com password=password
imconfutil set-prop -c InstantMessaging_home/config/iim.conf.xml smpp.bind_id=test
smpp.bind_password=password smpp.sender_id=test
smpp.smsg_ip_address=test.example.com smpp.smsg_port=2775
smpplib.jid=smpplib.example.com smpplib.password=password msgw.iim_
server=foo.example.com:5269
msgw.imadmin.enable=true
```

Example: Configuring the SMS Gateway and Instant Messaging Server on Different Hosts

In this example, the SMS gateway is configured on **im-1** and the Instant Messaging server is on **im-2**.

On host im-1:

```
imconfutil set-prop -c InstantMessaging_home/config/iim.conf.xml smpp.bind_id=test
smpp.bind_password=password smpp.sender_id=test smpp.smsg_ip_
address=test.example.com smpp.smsg_port=2775 smpplib.jid=smpplib.example.com
smpplib.password=password msgw.iim_server=bar.example.com:5269
msgw.imadmin.enable=true
```

On host im-2:

```
imconfutil add-component msggateway jid=smpplib.example.com password=password
```

Note: The value of the **jid** and **password** properties provided in the **add-component** command must be the same as the values that you define for the **smpplib.jid** and **smpplib.password** properties.

Configuring the SMS Gateway by Using the configure Utility

To configure the SMS gateway by using the **configure** utility:

1. Ensure that Instant Messaging Server has been installed.
2. Use [Table 21-1](#) and [Table 21-2](#) to determine the values to use.
3. Run the **configure** utility.

```
configure
```

4. Perform the following tasks in the configure panel.
 - a. Select the **Enable SMS Gateway** option by typing **yes**.
 - b. Select the **Enable Local Component** option by typing **yes**.

If you select this option, you can administer the SMS gateway by using the **imadmin** command. For example, to start the SMS gateway, you can enter **imadmin start sms-gateway**. You can also start the gateway by typing **imadmin start**.

5. Enter the XMPP server host name.

You can configure Instant Messaging Server and the SMS gateway on the same host or on different hosts. If you choose to configure the gateway for a remote

Instant Messaging server, specify the remote server host name. The default host name is the name of the local host.

6. Enter the port number.

The default value is the port number that you specify for the XMPP server. For example, if the XMPP server port is 5269, enter **5269**.

7. Enter the bind ID of the SMSC at the **ESME System Id** prompt.
8. Enter the SMSC bind password at the **ESME System Password** prompt.
9. Enter the IP address or the FQHN (Fully Qualified Host Name) of the SMSC at the **SMSC Host address** prompt.
10. Enter the SMSC port number at the **SMSC port** prompt. The default port number is **2775**.
11. Enter the Sender ID at the **SMS Sender ID** prompt.

The sender ID is the ID with which you have registered to the SMSC. The SMSC always send a SMS with the sender ID that you specify here.

Client-Side Settings

The Instant Messaging server searches for the recipient phone number in the following order of precedence:

1. Phone number settings in user v-card of a third-party messaging client
2. LDAP setting in the mobile attribute of Directory Server
3. Phone number settings in the Instant Messaging client

If you use a third-party messaging client such as Psi, specify the phone settings in the user v-card. See the third-party messaging client documentation for the procedure about adding phone settings.

If you use Directory Server, add the recipient phone number in the LDAP mobile attribute. For more information about the Directory Server, refer to the Directory Server documentation at:

<http://www.oracle.com/technetwork/middleware/id-mgmt/documentation/index.html>

Starting and Stopping the SMS Gateway

You can start and stop the SMS gateway by using the **imadmin** command. Before starting the SMS gateway, ensure that the Instant Messaging service and the SMSC service are online.

- To start the SMS gateway, enter the following command:

```
imadmin start sms-gateway
```

- To stop the SMS gateway, enter the following command:

```
imadmin stop sms-gateway
```

- To check the status of the SMS gateway, enter the following command:

```
imadmin status sms-gateway
```

Using Calendar Pop-up Reminders

Oracle Communications Instant Messaging Server is integrated with Oracle Communications Calendar Server to provide automatic pop-up reminders to Instant Messaging Server users for both calendar events and tasks. This chapter describes how to use this feature.

About Pop-up Reminders

This section contains information about Calendar pop-up reminders.

Pop-up Reminders Operation

Users can receive Instant Messaging pop-up reminders for upcoming events and tasks on their calendars. To enable these pop-up reminders, the following must occur:

- You must configure Calendar Server to use a notification service (JMQ), and the Instant Messaging server to enable pop-up notifications.
- The end user must enable calendar reminders in Instant Messaging Server.

With pop-ups enabled, when an impending event or task nears, the alarm set in the notification service causes Calendar Server to send an email notification and Instant Messaging Server to display a pop-up reminder.

Pop-up Reminders Architectural Flow

If configured, Instant Messaging Server pop-up reminders follow this architectural flow:

1. The Instant Messaging Server JMS subscriber subscribes to Calendar server events and notifications in the notification service, either JMQ or ENS.
2. Calendar server publishes an event or task notification in text/xml or text/calendar format to the notification service.
3. The Instant Messaging Server JMS subscriber receives the calendar event or task notification and then generates a message in text/calendar format.
4. The Instant Messaging server sends the message to the calendar owner, if the end user is online.
5. If the recipient is available, Instant Messaging Server generates an HTML pop-up reminder on the end user's desktop based on the message. If the recipient is not available, Instant Messaging Server discards the message.

Configuring Calendar Server and Instant Messaging Server to use Pop-ups

See "[Configuring Calendar Agent with Calendar Server](#)" for server-side configuration.

Configuring Calendar Pop-ups in a Server Pool

To configure Calendar pop-ups to work in a server pool deployment, you only need to configure one server's Calendar agent in the pool. A pop-up will be delivered for each configured Calendar agent in the pool.

Administering the Calendar Agent

The Calendar agent is an Instant Messaging Server component that provides pop-up functionality to Calendar and Instant Messaging users. In addition, using tools provided with Instant Messaging Server, you can start, stop, restart, or check the status of the Calendar agent as well as monitor its activity through log files. See "[Overview of Stopping, Starting, Refreshing, and Checking Instant Messaging Server Components](#)" for more information.

See "[Managing Logging for Instant Messaging Server](#)" for information about Calendar agent logs.

Configuring the Instant Messaging Server Calendar Agent

This chapter describes how to configure the Oracle Communications Instant Messaging Server calendar agent.

Configuring Calendar Agent with Calendar Server

This section describes how to configure Instant Messaging Server for Java Message Service (JMS) to support Oracle Communications Calendar Server Agent alerts. You can configure Instant Messaging Server either by running the **configure** command or by manually configuring the necessary properties with the **imconfutil** command. You do not need to configure Calendar Server itself to support calendar reminders.

Configuring Instant Messaging Server

No special steps are needed to configure Calendar Server to work with Instant Messaging Server. However, you do need to configure the Instant Messaging Server Calendar Agent. This section describes the two ways in which you can configure Instant Messaging Server, and properties that need to be configured.

Configuring Instant Messaging Server Calendar Agent with Calendar Server

To configure Instant Messaging Server Calendar Agent with Calendar Server:

1. Run the **configure** utility, for example:

```
/opt/sun/comms/im/sbin/configure --nodisplay
```

2. Respond to the Calendar Agent configuration prompts as follows:

```
Do you want to go back to previous panel [no]:no
Enable Calendar Agent [no]:y
Enable local component [no]:y
XMPP server hostname [hostname]: host_name
XMPP server port [port number]: port_number
JMQ Username: username
JMQ Password: password
Notification Server Hostname [localhost]: host_name
Notification Server Port [port number]: port_number
Topic: testTopic
```

3. After running the **configure** utility, use the **imconfutil** command to set the **contenttype** property.

For example:

```
/opt/sun/comms/im/sbin/imconfutil -c InstantMessaging_home/config/iim.conf.xml
set-prop agent-calendar.serveralarms.contenttype=text/plain
```

Manually Configuring Instant Messaging Server Calendar Agent with Calendar Server

If you did not configure the Calendar Agent while running the **configure** command, you can use the **imconfutil** command to manually set the properties.

1. The following example shows how to set the list of properties in "JMS and Calendar Agent Properties" by running the **imconfutil** command:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml add-component
id=calagent jid=calendar.domain password=secret
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop
agent-calendar.broker.address=host.domain:7676
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop
agent-calendar.consumer.topic=DavNotificationTopic
imconfutil -c InstantMessaging_home/config/iim.conf.xml set-prop
agent-calendar.imadmin.enable=true
imconfutil -c InstantMessaging_home/iim.conf.xml set-prop iim_
agent.agent-calendar.enable=true
imconfutil -c InstantMessaging_home/iim.conf.xml set-prop agent-calendar.iim_
server.host=host.domain
imconfutil -c InstantMessaging_home/iim.conf.xml set-prop agent-calendar.iim_
server.port=5269
imconfutil -c InstantMessaging_home/iim.conf.xml set-prop iim_agent.enable=true
imconfutil -c InstantMessaging_home/iim.conf.xml set-prop
agent-calendar.jid=calendar.domain
imconfutil -c InstantMessaging_home/iim.conf.xml set-prop
agent-calendar.password=secret
imconfutil -c InstantMessaging_home/iim.conf.xml set-prop
agent-calendar.serveralarms.contenttype=text/plain
```

2. Optional: Run the following command if your **imqbrokerd** is configured with a user name and password. You do not have to run the following command if Java Message Queue (JMQ) is being used from GlassFish Server.

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml add-jmqbroker id=jmq1
address=host_name:port user=guest password=guest
```

JMS and Calendar Agent Properties

[Table 23–1](#) list the JMS properties that you configure for Instant Messaging Server.

Table 23–1 JMS API Configuration Properties

Property	Default Value	Description
agent-calendar.broker.address	<i>JMQserver:port</i>	Specifies the host name and port on which the broker (JMQ) is running.
agent-calendar.consumer.topic	topic	Topic on which the calendar notifications are delivered.
agent-calendar.broker.user	guest	Specifies the broker user name.
agent-calendar.broker.password	guest	Specifies the password of the broker user name.

[Table 23–2](#) list the Calendar Agent properties that you configure for Instant Messaging Server.

Table 23–2 Instant Messaging Server Calendar Agent Configuration Properties

Property	Value	Description
<code>agent-calendar.imadmin.enable</code>	false	If set to true , you can start the agent-calendar by using the imadmin command.
<code>iim_agent.agent-calendar.enable</code>	false	If set to true , you can start the agent-calendar by using the imadmin command.
<code>agent-calendar.iim_server.host</code>	NA	Host name of the Instant Messaging server with which the agent calendar communicates.
<code>agent-calendar.iim_server.port</code>	NA	Port number of the Instant Messaging server with which the agent calendar communicates.
<code>iim_agent.enable</code>	true	Enables agents for Instant Messaging Server.
<code>iim_agent.enable</code>	true	Enables agents for Instant Messaging Server.
<code>agent-calendar.password</code>	NA	Password used by the Calendar Agent component to authenticate the Instant Messaging Server.
<code>agent-calendar.serveralarms.contenttype</code>	NA	Content type used to deliver calendar alerts. Can be text/plain or text/ical .

Notes About Configuring Calendar Server

- No additional steps are required to configure Calendar Server to work with Instant Messaging Server.
- By default, Calendar Server is configured with the **DavNotificationTopic** configuration parameter. Use **DavNotificationTopic** as the value for the **agent-calendar.consumer.topic** property.
- GlassFish Server uses Java Message Queue on port 7676 independently of Messaging Server and Calendar Server. A single host installation of Calendar Server might see problems with port 7676 being occupied if GlassFish Server is already installed on that host. To avoid this problem, edit the `/etc/imq/imqbrokerd.conf` file in the Oracle Solaris default path and set the **ARGS=-port** parameter to a free port.
- **imqbrokerd** should be up and running for Calendar Server alerts to work when configured with Java Message Queue.

Displaying Availability Based on Calendar Entries

This chapter describes how to display availability for Oracle Communications Instant Messaging Server based on entries in a user's calendar.

Overview of Displaying Instant Messaging Availability Based on Calendar Entries

Calendar availability is a feature that makes it possible to display a user's instant messaging availability based on the user's calendar schedule. For example, if calendar availability is enabled and user-A's calendar shows that user A has a meeting from 2:00 PM to 3:00 PM, Instant Messaging Server can inform other users that user A is busy and not available.

To use the calendar availability feature, you must:

1. Use the **imconfutil** command to enable calendar availability (see "[Enabling Instant Messaging Availability Based on Calendar Entries](#)").
2. Configure one or more Java Message Queue (JMQ) brokers for providing calendar information to Instant Messaging Server (see "[Configuring Java Message Queue Brokers for Calendar Availability](#)").

Enabling Instant Messaging Availability Based on Calendar Entries

To enable the use of calendar entries for displaying instant messaging availability, use the **imconfutil set-property** command to set the **agent-calendar.presence.enable** property to **true**:

```
imconfutil set-property agent-calendar.presence.enable=true
```

To disable the calendar availability feature once it has been enabled, set **agent-calendar.presence.enable** to **false**.

Configuring Java Message Queue Brokers for Calendar Availability

When the calendar availability is enabled, multiple calendar servers can send Java Message Queue (JMQ) notifications to an Instant Messaging server. For each calendar server, there needs to be a separate JMQ broker. To configure a JMQ broker, use the **imconfutil** commands in [Table 24-1](#).

Table 24–1 *imconfutil* Commands for Configuring a JMQ Broker

imconfutil Command	Description
add-jmqbroker	<p>Adds a JMQ broker for handling calendar availability notifications.</p> <p>Parameters:</p> <p><i>id</i>: A user-specified identifier for the added broker.</p> <p><i>address</i>: The host name and port (<i>hostname:port</i>) of the JMQ publisher that the calendar-agent broker communicates with. The JMQ publisher is a part of Calendar Server.</p> <p><i>user</i>: The user name for the broker to use in connecting to Instant Messaging Server.</p> <p><i>password</i>: The password for the broker to use in connecting to Instant Messaging Server.</p> <p>Example:</p> <pre>imconfutil -c InstantMessaging_home/config/iim.conf.xml add-jmqbroker id=broker1 address=jmqbroker.example.com:7676 user=thisjmqbroker password=zyxw</pre>
delete-jmqbroker	<p>Deletes a JMQ broker.</p> <p>Parameter:</p> <p><i>id</i>: The ID of the broker to be deleted.</p>
list-jmqbrokers	<p>Lists the IDs of all JMQ brokers. The properties of an individual broker can then be obtained using the imconfutil get-jmqbroker-prop command.</p> <p>Parameters: none.</p>
set-jmqbroker-prop	<p>Sets one or more properties of a JMQ broker.</p> <p>Parameters:</p> <p><i>id</i>: The ID of the broker.</p> <p><i>property</i>: A property of the broker. To set more than one property, enter a space-separated list of <i>property=value</i> pairs. The properties you can set are:</p> <ul style="list-style-type: none"> ■ <i>address</i>: the host name and port (<i>hostname:port</i>) of the JMQ publisher that the calendar-agent broker communicates with. The JMQ publisher is a part of Calendar Server. ■ <i>user</i>: The user name for the broker to use in connecting to Instant Messaging Server. ■ <i>password</i>: The password for the broker to use in connecting to Instant Messaging Server.
get-jmqbroker-prop	<p>Gets the properties assigned to a JMQ broker.</p> <p>Parameter:</p> <p><i>Id</i>: The ID of The Broker.</p>

Using the Web Presence API

This chapter describes how to use and configure the Oracle Communications Instant Messaging Server Web Presence API.

About the Instant Messaging Server Web Presence API

Users of instant messaging are typically able to see a contact list showing whether individual contacts are available. Instant messaging users who are not contacts typically do not receive such *presence* information. Instant Messaging Server provides a Web Presence API that allows a Web application to obtain presence information from Instant Messaging Server and display it to users, independent of whether they are contacts. An example of where this is useful is an enterprise application that provides an employee listing with information such as employee name, department, title, location, and phone number. If the application provides availability information, it may help a user who looks up an employee to decide whether to make a phone call, send an email, or look for someone else to contact.

For a Web application to obtain presence information from Instant Messaging Server, it needs to use the new Web Presence API and the Web Presence API must be configured to allow use by the Web application. In addition, any Instant Messaging server that the Web Presence API communicates with must be configured to recognize the Web Presence API (see ["Configuring an Instant Messaging Server to Recognize the Web Presence API"](#)).

Web Presence API for Requesting Presence Information

When a user's instant messaging application starts and contacts Instant Messaging Server, it sends presence information about the user. A web application can then obtain this information from Instant Messaging Server by making an HTTP GET or HTTP POST request that uses the Web Presence API. The request can be for the availability of a single user or of multiple users.

web.xml File for the Web Presence API

A template deployment descriptor (**web.xml**) file for the Web Presence API is provided at the following location:

```
InstantMessaging_home/lib/presenceapi-web.xml.template
```

You can edit the file to add any custom servlets you have developed for handling requests to the Web Presence API.

HTTP GET Requests for Presence Information

GET requests for presence information have different formats when they are sent to an individual user or to multiple users. In both cases, by default, the response to a request for presence information is returned in a JSON object (see ["JSON Response to Requests for Presence Information"](#)).

GET Requests for Presence Information on an Individual User

To send a **GET** request for presence information on an individual user, send the request to **/presence/jid_or_email_address** as in the following example:

```
GET /presence/node@domain.com/resource
```

By default, the request returns a JSON object containing the presence information (see ["JSON Response to Requests for Presence Information"](#)).

GET Requests for Presence Information on Multiple Users

To send a **GET** request for presence information on more than one user, send a request in the following format:

```
GET /presence?request={JSON object}
```

where the JSON object has the following format:

```
[{"presence":{"type":"probe","to":"node@domain/resource"}}, {"presence":{"type":"probe","to":"node1@domain1/resource2"}}, {"presence":{"type":"probe","to":"node2@domain/resource3"}}]
```

In the request (see RFC 3921, XMPP: Instant Messaging and Presence):

- **type** is a parameter for the type of presence request.
- **to** is a parameter for the user's JID or email address.
- **node@domain/resource** is the user's JID or email address.

For more information, see *RFC 3921, XMPP: Instant Messaging and Presence* at:

<http://www.ietf.org/rfc/rfc3921.txt>

By default, the request returns a JSON object containing the presence information (see ["JSON Response to Requests for Presence Information"](#)).

HTTP POST Requests for Presence Information

A **POST** request for presence information has the following format:

```
POST /presence
request:JSON object
```

where the JSON object contains one or more presence requests, as in the following example:

```
POST /presence
request:[{"presence":{"type":"probe","to":"node@domain/resource"}}, {"presence":{"type":"probe","to":"node1@domain1/resource2"}}, {"presence":{"type":"probe","to":"node2@domain/resource3"}}]
```

In the example:

- **type** is a parameter for the type of presence request.

- **probe** is a request for a contact's current presence information.
- **to** is a parameter for the user's JID or email address.
- **node@domain/resource** is the user's JID or email address.

For more information, see *RFC 3921, XMPP: Instant Messaging and Presence*:

<http://www.ietf.org/rfc/rfc3921.txt>

By default, the request returns a JSON object containing the presence information (see "[JSON Response to Requests for Presence Information](#)").

JSON Response to Requests for Presence Information

By default, requests for presence information return an HTTP response whose payload is a JSON object that contains the parameters of an XMPP presence stanza. The following example shows a JSON object with presence information on three users:

```
{ "response": [ { "presence": { "show": "dnd", "from": "node@domain/resource"
" } }, { "presence": { "show": "away", "status": "temporarily unavailable
", "from": "node1@domain/resource" } }, { "presence": { "from": "node2@domain/resource" } } ] }
```

where:

- **show** is a parameter for showing a user's availability.
- **dnd** (Do Not Disturb) means the user is busy.
- **from** is a parameter for the JID or email address of the user sending the presence information.
- **node@domain/resource** is the JID or email address of the user.
- **status** is a parameter for a description of the user's availability status.
- **temporarily unavailable** is the user's availability status.

Configuring the Instant Messaging Server Web Presence API

For a Web application to obtain and display presence information, you must configure properties in the Web Presence API's configuration file. A template for the configuration file is installed in: *InstantMessaging_home/lib*. There is no required name or location for the configuration file.

[Table 25-1](#) lists Web Presence API configuration properties.

Table 25–1 Web Presence API Configuration Properties

Property	Default Value	Description
presenceapi.idtype	jid	Specifies whether HTTP requests for presence information contain a JID (idtype is jid) or an email address (idtype is email).
presenceapi.wait_time	10	The maximum length of time, in seconds, that the presence component waits to receive a response from an Instant Messaging server that contains presence information for an individual user. If no response is received within the time limit, the presence component returns a presence type of none as the user's availability.
presenceapi.log4j.config	None	<p>The location of the configuration file that Instant Messaging Server uses for Apache-log4j logging.</p> <p>See "Managing Logging for Instant Messaging Server" for information on Instant Messaging Server logging.</p> <p>For information on Apache-log4j logging, see the Apache website at:</p> <p>http://logging.apache.org</p> <p>At installation, a presenceapi_log4j.conf.template template file to use as the basis for a log4j configuration file is installed in <i>InstantMessaging_home/config</i>. Use the template file to create a log4j configuration file. There is no required name or location for the configuration file.</p>
presenceapi.config	None	A space-separated list of identifiers for the Instant-Messaging-Server deployments that the presence component can communicate with. Each identifier will be used as a prefix to presence-API configuration properties for the deployment. For example, given the identifier list ImDeploy1 ImDeploy2 , there will be separate sets of ImDeploy1.property properties and ImDeploy2.property properties. Each deployment in the list of identifiers must be separately configured to recognize communications from the Web Presence API (see " Configuring an Instant Messaging Server to Recognize the Web Presence API ").
identifier.presencepolicy	open	<p>Specifies the way to interpret the list of JID or email domains in the <i>identifier.domains</i> property, one of the following:</p> <ul style="list-style-type: none"> ■ open (the default value): The Server supports presence requests for users in all domains except those listed in <i>identifier.domains</i>. ■ closed: The Server supports presence requests for users only in the domains listed in <i>identifier.domains</i>.
identifier.domains	None	<p>A space separated list of JID or email domains:</p> <ul style="list-style-type: none"> ■ If the presenceapi.idtype property is set to jid, list JID domains; if presenceapi.idtype is set to email, list email domains. ■ If the <i>identifier.presencepolicy</i> property is set to open, a list of domains that are not supported and for which presence information will not be retrieved. ■ If the <i>identifier.presencepolicy</i> property is set to closed, a list of domains that are supported and for which presence information is retrieved.
identifier.hosts	None	A space-separated list of Instant Messaging servers and their ports (<i>hostname:port</i>) that make up the deployment identified by <i>identifier</i> .

Table 25–1 (Cont.) Web Presence API Configuration Properties

Property	Default Value	Description
identifier.componentjid	None	The JID that the Web Presence API uses in establishing a connection with the Instant Messaging server specified by <i>identifier</i> . When you configure Instant Messaging Server, you must enter this JID to identify the Web Presence API to the Server (see " Configuring an Instant Messaging Server to Recognize the Web Presence API "). It is recommended that the same JID be used with each Instant Messaging server the Web Presence API communicates with.
identifier.password	None	The password that the presence component uses in establishing a connection with Instant Messaging Server specified by <i>identifier</i> . When you configure Instant Messaging Server, you will need to enter this JID to identify the Web Presence API to the Server (see " Configuring an Instant Messaging Server to Recognize the Web Presence API "). It is recommended that the same password be used with each Instant Messaging server. To generate an encrypted password, use the Instant Messaging Server " passwordtool Command Reference ".
presenceapi.component.password.cipher.delegate	None	If you want to use an encrypted password, this property is required with a value of com.sun.im.tools.passwordtool.Crypto .
presenceapi.component.password.cipher	None	If you want to use an encrypted password, this property is required with a value of com.sun.im.tools.passwordtool.CommsClientCipher .
presenceapi.cache_size	10000	The size in MB of the logging cache. If cache_size is set to -1 , the size of the logging cache is unlimited.
presenceapi.log_timeout	30	The time interval in minutes for statistics logging. To enable, uncomment the line.

Configuring an Instant Messaging Server to Recognize the Web Presence API

The Web Presence API is an XMPP component that communicates with Instant Messaging Server. The Web Presence API must be configured to communicate with the Instant Messaging Server, and Instant Messaging Server must be configured to receive communications from an XMPP component. This requires using the **imconfutil** command to add the Web Presence API as a component of the Instant Messaging Server, as in the following example:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml add-component
id=presenceapi jid=presenceapi.example.com password=drowp broadcastpresence=true
```

where:

- **jid** must be set to the same value as the Web Presence API's **ImServer1.componentjid** property.
- **password** must be set to the same value as the Web Presence API's **ImServer1.password** property.
- **broadcast** must be set to **true** to allow Instant Messaging Server to send presence information to the Web Presence API.

After adding the Web Presence API as a component to Instant Messaging Server, restart the server:

```
imadmin refresh server
```

Configuring and Testing the Web Presence API

This section provides steps for configuring and trying out the Web Presence API before you have done more extensive application development. The steps cover:

- Configuring and deploying the Web Presence API on a local machine.
- Configuring Instant Messaging Server for the Web Presence API and restarting the Server.
- Logging a user in to Instant Messaging Server through an XMPP client.
- Sending a GET request for the user's presence to Instant Messaging Server and viewing the response.

To configure and test the functionality of the Web Presence API:

1. Use the **iwadmin** command to create a ZIP file for the Web Presence API that you can move to another machine or keep locally:

```
iwadmin generatezip presenceapi -c /local/presenceAPI/config/presenceapi.conf  
-d /tmp/presenceapi.zip
```

where:

- The **-c** parameter specifies the future location of the configuration file for the Web Presence API, once the ZIP file is unzipped.
 - The **-d** parameter specifies a destination directory and file name for the ZIP file. The directory must exist before you generate the ZIP file.
 - The ZIP file contains:
 - A deployable WAR file for the Web Presence API
 - A template file (presenceapi.conf.template) to use for creating a Web Presence API configuration file.
 - A template file, **presenceapi_log4j.conf.template**, for configuring a log4j log file.
 - The Instant Messaging Server **password** tool. See "[passwordtool Command Reference](#)" for more information.
2. Extract the contents of the ZIP file to the directory that you want to use as your Web Presence API configuration directory, for example, **/local/presenceAPI**.

```
unzip /tmp/presenceapi.zip -d /local/presenceAPI
```

3. To create a configuration file for the Web Presence API, copy the configuration file template, **presenceapi.conf.template**, to **presenceapi.conf**.
4. In **presenceapi.conf**, edit the following lines according to the instructions below them.

```
presenceapi.config=default  
default.presencepolicy=open  
default.domains=DOMAINS_LIST  
default.hosts=HOSTS_LIST  
default.componentjid=COMPONENT_JID  
default.password=ENCRYPTED_COMPONENT_PASSWORD  
presenceapi.log4j.config=LOG4J_CONFIG_FILE  
# Comment out the following options if you are not using an encrypted password.  
presenceapi.component.password.cipher.delegate=DELEGATE_CLASS  
presenceapi.component.password.cipher=CIPHER_CLASS
```

For information about the configuration properties above, see [Table 25-1](#).

- a. Leave the first two entries, **presenceapi.config=default** and **default.presencepolicy=open** as they are.

The first property identifies a deployment of Instant Messaging servers as default.

The second property sets the deployment to provide presence information for contacts in all domains, except for domains that are listed with the **default.domains** property.

- b. Delete **default.domains=DOMAINS_LIST**. There is no need to restrict the domains open to presence requests in this example.
- c. Set **default.hosts** to a space-separated list of the Instant Messaging servers and their ports (*hostname:port*) that make up the default deployment. For example:

```
default.hosts=ImServer1.example.com:5269 ImServer2.example.com:5269
```

- d. Set **default.componentjid** to the JID of the Web Presence API in its Web server. The same JID must be entered when you use the command to configure Instant Messaging Server at a later step. Example:

```
default.componentjid=presenceapi.example.com
```

- e. Set **default.password** to the password you want to use in connecting to Instant Messaging Server. The same password must be entered when you use the command to configure Instant Messaging Server at a later step. Example:

```
default.password=asdfjkl;
```

- f. Set **presenceapi.log4j.config** to the absolute path of the log4j configuration file. For example:

```
set presenceapi.log4j.config=/local/presenceAPI/config/presenceapi_log4j.conf
```

5. Deploy the Web Presence API WAR file.

- GlassFish Server: Use the **asadmin** command to deploy the Web Presence API WAR file (you will need Administrator privileges), for example:

```
asadmin deploy /local/presenceAPI/presenceapi.war
```

- Oracle WebLogic Server: use WebLogic console to deploy the Web Presence API WAR file.

6. Use the **imconfutil** command to add the Web Presence API as an XMPP Web component to the Instant Messaging server identified as ImServer1:

```
imconfutil -c InstantMessaging_home/config/iim.conf.xml add-component  
id=presenceapi jid=presenceapi.example.com password=asdfjkl;  
broadcastpresence=true
```

where:

- **jid** must be set to the same value as the Web Presence API's **ImServer1.componentjid** property.
- **password** must be set to the same value as the Web Presence API's **ImServer1.password** property.

- **broadcast** must be set to **true** to allow Instant Messaging Server to send presence information to the Web Presence API.
7. Restart Instant Messaging Server using the **imadmin** command. This is necessary because of the configuration changes you made to the Server.

```
imadmin refresh server
```
 8. In preparation for the next step, sending a request for presence information, get the JID of a user that is logged-in to Instant Messaging Server through an XMPP client. If you have an instant messaging account on an XMPP client that uses the Instant Messaging Server, and your XMPP client is running, you can use your own JID.
 9. Send a **GET** or **POST** request for presence information to the Web Presence API, using the JID from the previous step.
 - You can send a **GET** request manually by entering it directly in the address bar of a browser and pressing Return. The following is an example of such a request. Note the / that follows the JID. Without it, the JID is interpreted as a file name and the request fails.

```
http://imhost@pythia.com:8080/presenceapi/presence/wxyz@example.com/
```

If the request is successful, a JSON object with presence information is displayed in the browser, as in the following example:

```
{"response":[{"presence":{"from":"wxyz@example.com/1379305740520"}}]}
```
 - You can send the request from a JavaScript, in which case the response is returned to the script as a JSON object.
 10. Verify that the response received is what was expected.

Configuring the Instant Messaging Server Web Presence API

This chapter describes how to configure the Oracle Communications Instant Messaging Server Web Presence API. The Web Presence API makes it possible for a Web application to obtain presence information from Instant Messaging Server and display it to users, independent of whether they are instant-messaging contacts. See ["Using the Web Presence API"](#) for information on using the Web Presence API.

Configuring the Web Presence API

To configure the Web Presence API, set configuration properties as described [Table 26–1](#).

Table 26–1 Web Presence API Configuration Properties

Property	Default Value	Description
<code>presenceapi.idtype</code>	<code>jid</code>	Specifies whether HTTP requests for presence information contain a JID (<code>idtype</code> is <code>jid</code>) or an email address (<code>idtype</code> is <code>email</code>).
<code>presenceapi.wait_time</code>	10	The maximum length of time, in seconds, that the presence component waits to receive a response from Instant Messaging Server that contains presence information for an individual user. If no response is received within the time limit, the presence component returns a presence type of none as the user's availability. The default value is 10 seconds if there is no entry for this property in the Presence API configuration file. In the Presence API configuration template, the value is set to 15 seconds. Thus, if you use the template, then the default is 15. If you create the configuration file from scratch, the default is 10 seconds.
<code>presenceapi.log4j.config</code>	None	The location of the configuration file that Instant Messaging Server uses for Apache-log4j logging. See "Managing Logging for Instant Messaging Server" for information on logging. For information on Apache-log4j logging, see the Apache website at: http://logging.apache.org At installation, a <code>presenceapi_log4j.conf.template</code> template file to use as the basis for a log4j configuration file is installed in <code>InstantMessaging_home/config</code> . Use the template file to create a log4j configuration file. There is no required name or location for the configuration file.

Table 26–1 (Cont.) Web Presence API Configuration Properties

Property	Default Value	Description
presenceapi.config	None	<p>A space-separated list of identifiers for Instant Messaging Server deployments that the presence component can communicate with. Each identifier is used as a prefix to presence-API configuration properties for the deployment.</p> <p>For example, given the identifier list <code>ImDeploy1 ImDeploy2</code>, there are separate sets of ImDeploy1.property properties and ImDeploy2.property properties. Each deployment in the list of identifiers must be separately configured to recognize communications from the Web Presence API (see "Configuring an Instant Messaging Server to Recognize the Web Presence API").</p>
identifier.presencepolicy	open	<p>Specifies the way to interpret the list of JID or email domains in the <i>identifier.domains</i> property, one of the following:</p> <ul style="list-style-type: none"> ▪ open (the default value): The Server supports presence requests for users in all domains except those listed in <i>identifier.domains</i>. ▪ closed: The Server supports presence requests for users only in the domains listed in <i>identifier.domains</i>.
identifier.domain	None	<p>A space separated list of JID or email domains:</p> <ul style="list-style-type: none"> ▪ If the presenceapi.idtype property is set to jid, list JID domains; if presenceapi.idtype is set to email, list email domains. ▪ If the <i>identifier.presencepolicy</i> property is set to open, a list of domains that are not supported and for which presence information will not be retrieved. ▪ If the <i>identifier.presencepolicy</i> property is set to closed, a list of domains that are supported and for which presence information is retrieved.
identifier.hosts	None	A space-separated list of Instant Messaging servers and their ports (<i>hostname:port</i>) that make up the deployment identified by <i>identifier</i> .
identifier.component.jid	None	The JID that the Web Presence API uses in establishing a connection with the Instant Messaging server specified by <i>identifier</i> . When you configure Instant Messaging Server, you will need to enter this JID to identify the Web Presence API to the Server (see "Configuring the Instant Messaging Server Web Presence API"). It is recommended that you use the same JID for each Instant Messaging server the Web Presence API communicates with.
identifier.password	None	The password that the presence component uses in establishing a connection with the Instant Messaging server specified by <i>identifier</i> . When you configure Instant Messaging Server, you will need to enter this JID to identify the Web Presence API to the Server (see "Configuring an Instant Messaging Server to Recognize the Web Presence API"). It is recommended that you use the same password for each Instant Messaging server. To generate an encrypted password, use the Instant Messaging Server password tool (see "passwordtool Command Reference").
presenceapi.component.password.cipher.delegate	None	If you want to use an encrypted password, this property is required with a value of com.sun.im.tools.passwordtool.Crypto .
presenceapi.component.password.cipher	None	If you want to use an encrypted password, this property is required with a value of com.sun.im.tools.passwordtool.CommsClientCipher .

Part III

Instant Messaging Server Reference

Part III contains the following chapters:

- [Configuration File and Directory Structure Overview](#)
- [Configuration Properties](#)
- [Instant Messaging Server APIs](#)
- [imadmin Command Reference](#)
- [imconfutil Command Reference](#)
- [iwadmin Command Reference](#)
- [passwordtool Command Reference](#)
- [XMPP and HTTP Gateway Configuration Parameters](#)

Configuration File and Directory Structure Overview

This chapter describes the configuration files you use to administer Oracle Communications Instant Messaging Server. The chapter also describes the Instant Messaging Server directory structure and the properties files used to store Instant Messaging Server operational data and configuration information.

Program Files

Program files include the native executable files, the library files in the **bin** or **lib** directory, the shell scripts in the **sbin** directory, the Java classes, and templates files in the **lib** directory.

Oracle Solaris Location of Program Files

Program files are located in the Instant Messaging Server installation directory. The default location of the installation directory is **/opt/sun/comms/im**.

Red Hat Linux and Oracle Linux Location of Program Files

Program files are located in the Instant Messaging Server installation directory. The default location of the installation directory is **/opt/sun/comms/im**.

Server Configuration Files

Server configuration files include the **iim.conf.xml** file and a subdirectory that contains all the server-wide access control files.

Note: Use the **imconfutil** command to make a configuration change. Never directly edit the **iim.conf.xml** file.

Oracle Solaris Location of Server Configuration Files

Server configuration files are located in the Instant Messaging Server configuration directory. The default location of the configuration directory is **/etc/opt/sun/comms/im/default/config**.

For convenience, the installer creates a symbolic link from **/etc/opt/sun/comms/im/default/config** to **/opt/sun/comms/im/config**.

In addition, if you create multiple instances of Instant Messaging Server, the name of the **/default** directory varies, depending on the instance. See the topic on creating multiple instances from a single Instant Messaging Server installation in *Instant Messaging Server Installation and Configuration Guide*.

Red Hat Linux and Oracle Linux Location of Server Configuration Files

Server configuration files are located in the Instant Messaging Server configuration directory. The default location of the configuration directory is **/etc/opt/sun/im/default/config**

For convenience, the installer creates a symbolic link from **/etc/opt/sun/im/default/config** to **/opt/sun/comms/im/config**.

In addition, if you create multiple instances of Instant Messaging Server, the name of the **/default** directory varies, depending on the instance. See the topic on creating multiple instances from a single Instant Messaging Server installation in *Instant Messaging Server Installation and Configuration Guide*.

Runtime Directory

The runtime directory contains Instant Messaging Server data. It includes the configurable directory for files generated by the server at runtime. It also includes the end user data in the data directory. Its **log** directory contains the server, multiplexor, Calendar agent, and XMPP service log files.

Oracle Solaris Location of the Runtime Directory

The default value of the runtime directory is **/var/opt/sun/comms/im/default**.

In addition, if you create multiple instances of Instant Messaging Server, the name of the **/default** directory varies, depending on the instance. See the topic on creating multiple instances from a single Instant Messaging Server installation in *Instant Messaging Server Installation and Configuration Guide*.

Red Hat Linux and Oracle Linux Location of the Runtime Directory

The default value of the runtime directory is **/var/opt/sun/im/default**.

In addition, if you create multiple instances of Instant Messaging Server, the name of the **/default** directory varies, depending on the instance. See the topic on creating multiple instances from a single Instant Messaging Server installation in *Instant Messaging Server Installation and Configuration Guide*.

Database Directory

If you are using a file-based property store, the database directory contains end user information such as the user and news channels directory. If you are using LDAP to store user data, the database directory is not used.

Oracle Solaris Location of the Database Directory

The default value for the Database Directory is **/var/opt/sun/comms/im/default/db**

In addition, if you create multiple instances of Instant Messaging, the name of the **/default** directory will vary depending on the instance. See the topic on creating

multiple instances from a single Instant Messaging Server installation in *Instant Messaging Server Installation and Configuration Guide*.

Red Hat Linux and Oracle Linux Location of the Database Directory

The default value for the Database Directory is `/var/opt/sun/im/default/db`

In addition, if you create multiple instances of Instant Messaging, the name of the `/default` directory will vary depending on the instance. See the topic on creating multiple instances from a single Instant Messaging Server installation in *Instant Messaging Server Installation and Configuration Guide*.

Instant Messaging Server Configuration File

Instant Messaging Server stores all configuration properties (formally called *options*) in the `iim.conf.xml` file. See "[Configuration Properties](#)" for more information on the properties and values stored in the `iim.conf.xml` file.

Configuration Properties

This chapter covers the Oracle Communications Instant Messaging Server configuration properties that are stored in the **iim.conf.xml** file.

iim.conf.xml File Location

Instant Messaging Server stores configuration settings in the **iim.conf.xml** file within the configuration directory (*InstantMessaging_home/config*). The default is: **/etc/opt/sun/comms/im/default/config/iim.conf.xml**.

If you created multiple instances of Instant Messaging Server, the name of the **/default** directory varies depending on the instance. See the topic on creating multiple instances from a single Instant Messaging Server installation in *Instant Messaging Server Installation and Configuration Guide*.

iim.conf.xml File Syntax

- Instant Messaging Server merges its multiple configuration files into one file, the **iim.conf.xml** file.
- A set of common properties is used across the Instant Messaging Server deployment or instance, including **instancedir**, **installdir**, and so on.
- Each component has its own configuration section in the file, for example, a **server** section, a **mux** section, and so on.
- Some configuration properties are complex, in the sense that they have more than one instance, and each instance has one or more keys.
- When you run the **imconfutil** command, the **iim.conf.xml** file is updated. You must refresh Instant Messaging Server for the new configuration settings to take effect.

Note: The **iim.conf.xml** file is initialized by the installation process and should be modified only as described in this documentation.

The new configuration system has backward compatibility. If support for any key is not available with **imconfutil**, or if a property is not found in the **iim.conf.xml** file, it uses the old **iim.conf** as a fallback. If you want to upgrade from releases prior to Instant Messaging Server 9, you must copy **im.conf-pre9** to **iim.conf** for the fallback to work.

Table 28–1 lists and describes the general configuration properties.

Table 28–1 General Configuration Properties

Property	Default Value	Description
iim.smtpserver	localhost	SMTP server to send mail to end users who have set the option for forwarding their messages as emails or to pagers.
iim.instancedir	/opt	The installation directory root.
iim.instancevardir	Solaris: /var/opt/sun/comms/im/default Red Hat Linux and Oracle Linux: /var/opt/sun/im/default	Sets the directory to contain runtime files, including the end-user profile database, logs, and other files created by the server and multiplexor at runtime. The name of the /default directory may vary if you created multiple instances of Instant Messaging Server.
iim.user	inetuser	The end-user name with which the server processes run.
iim.group	inetgroup	The group with which the server processes run.
iim.jvm.maxmemorysize	256	The maximum number heap size in MB the JVM running the server is allowed to use. Used to construct the -mx argument of the Java command.
iim.mail.charset	None	Specifies whether the headers of the mail are in ASCII and not encoded. It contains the name of the character set to use to encode the headers of the mail message sent out for offline alerts. For example: <code>iim.mail.charset=iso-2022-jp</code>
iim.jvm.command	None	The location of the Java Runtime Executable (JRE).
iim.policy.cachevalidity	10	Defines the cache validity interval (in seconds) for a single user's information. Instant Messaging Server saves the last date a single end-user's information was cached. If the end-user's information is accessed after the interval determined by this property, the server will recache the end user's information and reset the cache date on the LocalUser object.
iim.policy.modules	iim_ldap	By default, LDAP is used for policy storage.
iim.userprops.store	ldap	Indicates whether the user properties are in a user properties file or stored in LDAP. Only significant when the service definitions for the Presence and Instant Messaging services have been installed. Value can be one of the following: <ul style="list-style-type: none"> ▪ ldap (the default) ▪ file (not recommended) In Convergence, if Instant Messaging Server is configured with a user properties file, end users are unable to upload Avatars. When using Instant Messaging Server in Convergence, the iim.userprops.store property should be set to ldap , not file .

Table 28–1 (Cont.) General Configuration Properties

Property	Default Value	Description
<code>iim_server.db_path</code>	<code>iim.instancevardir/db</code>	File system location for the server to store user properties. Used by the propstore file.
<code>iim_server.scratch_directory</code>	<code>iim.instancevardir/scratch</code>	Specifies the directory in which Instant Messaging Server can store transient data.
<code>iim_jid.encoding.compat</code>	<code>true</code>	Controls whether the server uses the standard JID encoding scheme or an older scheme that is compatible with older server versions.
<code>iim_server.jvm.options</code>	None	Specifies the options to be passed to the Java VM when starting the server.
<code>iim_server.maxsessions</code>	Equal to <code>iim_mux.maxsessions</code> by default. If <code>iim_mux.maxsessions</code> is not set, then <code>iim_mux.maxsessions</code> is 5000.	The maximum number of client sessions that the server can handle. Needs to be equal to at least the sum of all the multiplexor maxsessions .
<code>iim_server.maxthreads</code>	50	The maximum number of threads used by the server.
<code>iim_server.memory.user.cache.count</code>	<code>false</code>	Specifies whether user cache count is enabled.
<code>iim_server.component.requiressl</code>	<code>false</code>	Specifies whether all component connections must be encrypted.
<code>iim_server.classpath</code>	Includes custom paths in the classpath of the server.	No description
<code>iim_server.conference.servicename</code>	<code>muc</code>	Sets the conference service name.
<code>iim_server.sasl.usemechanism.DIGEST-MD5</code>	<code>false</code>	Enables the DIGEST-MD5 authentication mechanism.
<code>iim_server.filter.enable</code>	<code>false</code>	Specifies whether the file filter providers should be enabled or disabled.
<code>iim_server.filters</code>	<code>false</code>	Specifies the list of file filter providers.
<code>iim_server.deliverofflinechat</code>	<code>false</code>	Determines whether the capability is on or off. To enable the feature for the entire deployment, set the <code>iim_server.deliverofflinechat</code> property to true , and do not set <code>deliverofflinechat.domain</code> property. To disable the feature for the entire deployment, set the <code>iim_server.deliverofflinechat</code> property to false , and do not set <code>deliverofflinechat.domain</code> property.
<code>iim_server.peerpingtimeout</code>	300	Specifies the time in seconds that the server waits for a ping response from the peer. If set to a value less than or equal to 0, ping between the peers is disabled.
<code>iim_server.enablegroupsinconference</code>	<code>false</code>	Enables and disables the use of groups in multiuser chat rooms.

Table 28–1 (Cont.) General Configuration Properties

Property	Default Value	Description
deliverofflinechat.domain	None	Used to blacklist or whitelist a domain. To blacklist a domain, set the iim_server.deliverofflinechat property to true , and set the deliverofflinechat.domain property to the list of domains to be blacklisted. To whitelist a domain, set the iim_server.deliverofflinechat property to false , and set the deliverofflinechat.domain property to the list of domains to be whitelisted.
deliverofflinechat.maxsize	50	Determines the maximum queue size related to the Receiver.
iim_ldap.sasl.mechanism.factories	None	Specifies the mechanism that will be used for authentication.
iim_ldap.userpasswordattr	userpassword	Specifies the field that should be used for authentication. By default, userpassword is used for authentication.
iim_server.clientping.timeout	Disabled	Specifies time in seconds the server waits for a ping response from the client, before closing the connection. Use a negative value to disable the timeout.
iim_server.msg_archive.maxstanzas	10	Sets the maximum number of stanzas to be retrieved from the archive store.
iim_server.pubsub.customnode.enable	false	Enables or disables the pub-sub node feature.
iim_server.pubsub.customnode.offlinemessage	None	Used to set the topic name for the pub-sub node.
iim_server.tls.enabledprotocols	The default value depends on the JDK version. If empty, it takes the JDK built-in default enabled protocols.	Enables or disables the TLSv1, TLSv1.1, and TLSv1.2 protocols.
iim_server.carbon.enable	false	Enables or disables message carbons.
iim_server.hosteddomains.allowcrossdomainsaccess	false	Enables inter-hosted domain communication.
iim_server.hosteddomains.activelist	None	When inter-hosted domain communication is enabled, specifies the active list as whitelist , blacklist , or none .
iim_server.hosteddomains.whitelist	None	Specifies the whitelist of the domains for inter-hosted domain communication.
iim_server.hosteddomains.blacklist	None	The blacklist of the domains for inter-hosted domain communication.
iim_server.jdbc.connection.username	None	Sets the user name of the Oracle Database user.
iim_server.jdbc.connection.password	None	Sets the password of the Oracle Database user.

Table 28–1 (Cont.) General Configuration Properties

Property	Default Value	Description
<code>iim_server.jdbc.connection.initialCapacity</code>	10	The initial number of data base connections that are created when the pool is started.
<code>iim_server.jdbc.connection.maxCapacity</code>	10	The maximum number of active connections that can be allocated from this pool at the same time.
<code>iim_server.storage.providers</code>	None	Specifies the class names of the storage providers (comma separated).
<code>iim_server.conference.history.persist</code>	false	Specifies whether conference history is stored in persistent storage.
<code>iim_server.inboundpacketfilters</code>	Disabled	Specifies the PacketFilter implementation classes for inbound packets.
<code>iim_server.ext_disco.provider</code>	None	Specifies the provider class for the External Service Discovery plugin.

Table 28–2 lists and describes the properties used by Instant for LDAP, user registration, and user source configuration.

Table 28–2 LDAP, User Registration, and Source Configuration Properties

Property	Default Value	Description
<code>iim_ldap.host</code>	localhost:389	LDAP server name and port used by Instant for end-user authentication.
<code>iim_ldap.searchbase</code>	o=internet	The string used as base to search for the end users and groups on the LDAP server.
<code>iim_ldap.usergroupbinddn</code>	None (the server performs anonymous searches)	Specifies the DN to use to bind to the LDAP server for searches.
<code>iim_ldap.usergroupbindcred</code>	None (the server performs anonymous searches)	Specifies the password to use with the <code>iim_ldap.usergroupbinddn</code> DN for LDAP searches.
<code>iim_ldap.loginfilter</code>	<code>(&(!(objectclass=inetorgperson)(objectclass=webtopuser))(uid={0}))</code>	Search filter used during end-user login. The entire filter is entered as one line.
<code>iim_ldap.userbrowsefilter</code>	<code>(objectclass=inetorgperson)</code>	Specifies LDAP filter to be applied when browsing users.
<code>iim_ldap.usergroupbyidsearchfilter</code>	<code>(!((&(objectclass=groupofuniqueNames)(uid={0}))(&(!(objectclass=inetorgperson)(objectclass=webtopuser))(uid={0}))))</code>	The search filter used to search for end users and groups in the directory, under the base specified by ID. The entire filter is entered as one line.
<code>iim_ldap.usergroupbynamefilter</code>	<code>(!((&(objectclass=groupofuniqueNames)(cn={0}))(&(!(objectclass=inetorgperson)(objectclass=webtopuser))(cn={0}))))</code>	The search filter used to search for end users and groups in the directory, under the base specified by name.
<code>iim_ldap.usergroupbymailsearchfilter</code>	<code>(!((&(objectclass=groupofuniqueNames)(mail={0}))(&(objectclass=inetorgperson)(mail={0}))))</code>	The search filter that returns a group, given a mail address.

Table 28–2 (Cont.) LDAP, User Registration, and Source Configuration Properties

Property	Default Value	Description
iim_ldap.allowwildcardinuid	false	Determines if wildcards should be enabled for UIDs while performing a search. As most directory installations have UIDs indexed for exact searches only, the default value is False . Setting this value to True can impact performance unless UIDs are indexed for substring search.
iim_ldap.userclass	inetOrgPerson,webtopuser	The LDAP class that indicates that an entry belongs to an end user.
iim_ldap.groupclass	groupOfUniqueNames	The LDAP class that indicates that an entry belongs to a group.
iim_ldap.groupbrowsefilter	(&(objectclass=groupofuniquecnames)(cn={0}))	The search filter used to browse all groups in the directory under the specified search base.
iim_ldap.searchlimit	40	Maximum number of entries to be returned by a search. A value of -1 means search is disabled on this server and a value of 0 indicates unlimited search.
iim_ldap.resyncntime	720	Maximum time in seconds that data fetched from LDAP is held before resyncing.
iim_ldap.userdisplay	cn	LDAP attribute to use for display name of end users.
iim_ldap.groupdisplay	cn	LDAP attribute to use for display name of groups.
iim_ldap.useruidattr	uid	LDAP attribute used as end users' UID.
iim_ldap.groupmemberattr	uniquemember	LDAP attribute that gives the list of members of a group.
iim_ldap.usermailattr	mail	LDAP attribute that should contain end users' provisioned email addresses. Used when the email message is sent to an offline end user.
iim_ldap.usermobileattr	mobile	LDAP attribute that contains end users' mobile phone numbers.
iim_ldap.groupmemberattr	uniquemember	LDAP attribute that contains the group member DNs.
iim_ldap.groupmemberurlattr	memberurl	The membership attribute of a dynamic group, which contains the LDAP filter or the LDAP URL.
iim.register.enable	None	If TRUE , the server allows new Instant Messaging Server end users to register themselves (add themselves to the directory).
iim_ldap.register.basedn	None	If self-registration is enabled, the value of this property is the DN of the location in the LDAP directory in which person entries are stored. For example: "ou=people,dc=siroe,dc=com"
iim_ldap.register.domain	None	The domain to which new users will be added. For example, directory.siroe.com .
iim_ldap.firstnameattr	givenname	The LDAP attribute that stores the user's first name.
iim_ldap.user.attributes	None	The LDAP attribute that contains the list of custom attributes from the LDAP user entry.
iim_ldap.group.attributes	None	The LDAP attribute that contains the list of custom attributes from the LDAP group entry.

Table 28–2 (Cont.) LDAP, User Registration, and Source Configuration Properties

Property	Default Value	Description
iim_ldap.lastnameattr	sn	The LDAP attribute that stores the user's last name.
iim_ldap.managedroleobjectclasses	nsManagedRoleDefinition	The LDAP object class that represents managed-role objects.
iim_ldap.usessl	false	Specifies whether to use SSL when connecting to the primary LDAP server.
iim_ldap.schema1.domain_config_root	None	Specifies the base DN for looking up schema 1 domains.
iim_ldap.schema2.domain_config_root	None	Specifies the base DN for looking up schema 1 domains.
iim_ldap.debugPool	false	Enables extra debugging log messages for LDAP pool failover.
iim_ldap.groupchatstorage.queuesize	10000 if no value set, 1000 if value set and is less than 1000.	Specifies queuesize for group chat.
iim_ldap.plaintextpasswords	false	Specifies whether the passwords in LDAP are stored in clear text

Table 28–3 lists and describes the logging configuration properties for log4j-based logging.

Table 28–3 Logging Configuration Properties

Property	Default Value	Description
agent-calendar.log4j.refresh	60	Specifies in seconds how often the calendar agent rereads the log4j configuration.
agent-calendar.log4j.refresh	<i>InstantMessaging_cfg</i>	Specifies the location and name of the log4j configuration file. If no value exists for this property, the logger will look for log4j.conf in <i>InstantMessaging_cfg</i> . If the logger does not find log4j.conf in <i>InstantMessaging_cfg</i> , it uses the property-based logging method, instead of log4j.
iim_mux.log4j.refresh	60	Specifies in seconds how often the multiplexor rereads its log4j configuration.
iim_server.log4j.refresh	60	Specifies in seconds how often the server rereads its log4j configuration.
iim_wd.log4j.refresh	60	Specifies in seconds how often the watchdog rereads its log4j configuration.
iim_smpbind.log4j.refresh	60	Specifies in seconds how often the SMS gateway rereads its log4j configuration.

Table 28–4 lists and describes the general configuration properties.

Table 28–4 General Instant Messaging Server Configuration Properties

Property	Default Value	Description
iim_server.autosubscribe	false	Indicates whether subscriptions are automatically authorized by the server. The possible values are TRUE and FALSE . If TRUE , subscribe requests are automatically followed by a subscribed response generated by the server. The server then sends the modified roster to the subscriber and the user the subscriber added as a contact. The user and the contact must be in the same domain to use this feature.
iim_server.domainname	<i>host's domain name</i>	The logical Instant Messaging Server domain name you want this server to support. This is the name that is used by other servers in the network to identify this server. It is also the name used by this server to identify its end users to other servers. This is not necessarily the Fully Qualified Domain Name of the system running the Instant Messaging Server. For example, if the system iim.xyz.com is the only Instant Messaging Server instance for a company xyz.com , then the domain name is likely to be xyz.com .
iim_server.port	5269	IP address and port for the server to bind to, and to listen for connections from other servers. IP address setting is useful for multi-homed machines when you want to use only one particular IP address. If no IP address is listed, this indicates a value of INADDR_ANY on localhost .
iim_server.useport	true	Indicates whether the server should listen on the server-to-server communication port. The possible values are TRUE and FALSE . If TRUE , the server listens on the port defined by <i>iim_server.port</i> or on port 5269, if that is not explicitly defined.
iim_server.clienttimeout	15	Specifies the time, in minutes, before the server discards client connections that are no longer active. For example, when a machine is turned off. The minimum accepted value is 5.
iim_server.usesso	The default value is 1 , if you chose to leverage an Access Manager deployment for SSO when you ran the configure utility. Otherwise, the default value is 0 .	Tells the server whether to depend on the SSO provider during authentication. An SSO provider is a module the server uses to validate a session ID with a SSO service. The Access Manager Session API provides Instant Messaging Server with the ability to validate session IDs sent by the client. The value can either be 0 or 1 . Use SSO provider only without attempting LDAP authentication even when the SSO validation fails. The iim_server.usesso property is used with the iim_server.ssoprotider property.
iim_server.ssoprotider	None	Specifies the class implementing the com.sun.im.provider.SSOProtider interface. If iim_server.usesso is not equal to 0 and this option is not set, the server uses the default Access Manager-based SSO Provider.
iim.policy.modules	The default value is identity , if you chose to leverage an Access Manager deployment for policy when you ran the configure utility. Otherwise, the default value is iim_ldap .	If the value is identity , indicates that Access Manager is used for policy storage. If the value is iim_ldap , directory is used.

Table 28–4 (Cont.) General Instant Messaging Server Configuration Properties

Property	Default Value	Description
<code>iim.userprops.store</code>	<code>file</code>	If the value is file , indicates that the user properties are stored in a user properties file. If the value is ldap , directory is used.
<code>iim_server.msg_archive</code>	<code>false</code>	Specifies whether the archive provider should be enabled or disabled. Set this value to false to disable all archiving. Set the value to true to enable all archiving, including email, and any custom archive provider you want to use.
<code>iim_server.msg_archive.provider</code>	None	Contains the list of archive providers. Allows multiple values and each value is separated by a comma (.). If you are using email archiving, the value should be com.ipplanet.im.server.EmailIMArchive .
<code>iim_server.msg_archive.auto</code>	<code>false</code>	Specifies whether messages are automatically archived.
<code>iim_server.enable</code>	<code>true</code>	This value determines whether Instant Messaging Server is enabled. Set to false to enable the Instant Messaging multiplexor.
<code>iim_server.certnickname</code>	Server-Cert	This value should contain the name of the certificate you entered while installing the certificate. The certificate name is case-sensitive.
<code>iim_server.sslkeystore</code>	None	Contains the relative path and file name for the server's Java keystore (JKS), for example, <i>InstantMessaging_cfg/server-keystore.jks</i> .
<code>iim_server.keystorepasswordfile</code>	<code>sslpassword.conf</code>	This value should contain the relative path and the name of the file containing the password for the key database. This file should contain the following line: <code>Internal (Software) Token:password</code> where <i>password</i> is the password protecting the key database.
<code>iim_server.requiressl</code>	<code>false</code>	If true , the server terminates any connection that does not request a TLS connection after the initial stream session is set up. This includes connections from clients, other servers, and server components, such as the XMPP/HTTP Gateway and agents, except the multiplexor.
<code>iim_server.trust_all_cert</code>	<code>false</code>	If this value is true , the server trusts all certificates and also adds the certificate information into the log files.
<code>iim_server.dialback.key</code>	None	Defines a static dialback key.

Table 28–4 (Cont.) General Instant Messaging Server Configuration Properties

Property	Default Value	Description
iim_server.conversion.provider	None	Contains the list of Message Conversion Providers to be used for message conversion. It allows multiple values, with each value separated by a comma (,).
iim_server.conversion.external.command	None	Contains the external command used for message conversion. The iim_server.conversion.external.command property is used only in the default implementation of the Message Conversion API, which is com.ipplanet.im.server.ExternalDocumentConverter . This implementation invokes an external third party application. To use this property, you must set the class com.ipplanet.im.server.ExternalDocumentConverter as the provider class, and set the property iim_server.conversion.external.command="your_external_app_command %i %o" , where %i and %o will automatically be replaced by the actual input/output file names generated dynamically by ExternalDocumentConverter . For example, if a conversion application is located at /usr/local/bin/convert , use iim_server.conversion.external.command="/usr/local/bin/convert %i %o" .
iim_server.conversion	iim_server.conversion	Specifies whether message conversion should be enabled. Also specifies whether the configured list of Message Conversion Providers should be used for message conversion.

Multiple Server Configuration Properties

To enable communication between multiple Instant Messaging Server instances in your network, you must configure your server to identify itself with the other servers and identify itself with each coserver, or cooperating server, which has a connection to your server. The coserver identifies itself with its Instant Messaging Server domain name, host and port number, server ID, and password.

Each cooperating server is given a symbolic name, which is a string consisting of letters and digits, for example, **coserver1**. Using the symbolic naming convention you can specify multiple servers.

When Instant Messaging servers are configured in this manner, you can form a larger Instant Messaging Server community. Therefore, end users on each server can do the following:

- Communicate with end users on every other server
- Use conferences rooms on other servers
- Subscribe to news channels on other servers (subject to access privileges)

[Table 28–5](#) lists and describes the multiple server configuration properties.

Table 28–5 Multiple Server Configuration Properties

Property	Default Value	Description
<code>iim_server.serverid</code>	None	String used by this server to identify itself to all other servers.
<code>iim_server.password</code>	None	Password used by this server to authenticate itself to all other servers.
<code>iim_server.federation.policy</code>	None	To enable open federation, set to open . To disable open federation, set to closed . For example: <code>iim_server.federation.policy = "open"</code> By default, <code>iim_server.federation.policy</code> is not enabled.
<code>iim_server.federation.exceptions</code>	None	Specifies a blacklist of domains where federation is denied.
<code>iim_server.c2s.requiresssl</code>	false	Specifies whether all client-to-server communication must be encrypted.
<code>iim_server.conference.history.maxstanzas.default</code>	10	Specifies the maximum history stanzas sent to the client by default.
<code>iim_server.s2s.requiresssl</code>	false	Specifies whether all server-to-server communications must be encrypted.
<code>iim_server.conference.history.maxstanzas</code>	0	Specifies the maximum history stanzas stored by the server in memory.
<code>iim_server.conference.history.persist</code>	false	Specifies whether the server stores conference history in persistent storage.
<code>iim_server.conference.distributeall</code>	false	Specifies whether the server distributes conference room messages to all peers in a server-pool.

Shoal Configuration Properties

You can use the Shoal clustering framework to automatically discover and add peer servers in a server pool. [Table 28–6](#) lists and describes the shoal configuration properties.

Table 28–6 Shoal Configuration Properties

Property	Default	Required	Description
<code>iim_server.peer.autodiscover</code>	false	No	Enables auto-discovery using Shoal. It is recommended to delete all static co-server definitions before setting this to true.
<code>iim_server.serverid</code>	None	Yes	The ID that uniquely identifies the server instance within the pool. It could be an identifier such as server1 , or a host name.
<code>iim_server.password</code>	None	Yes	The password that is shared across the pool and enables identification of members of one pool from the other. Also ensures that unidentified members of a Shoal group are not able to join the pool.

Table 28–6 (Cont.) Shoal Configuration Properties

Property	Default	Required	Description
<code>iim_server.hostname</code>	<code>local-hostname:5269</code>	No	The connection string that the other pool members can establish connections with. It is the host name and port of the specified server.
<code>iim_server.pool.groupname</code>	<code>iim.server.pool</code>	No	The Shoal group name that the peers will attempt to join. You will need to change the default only if multiple clusters of peer servers need to run on the same subnet.
<code>iim_server.peer.conferences.usep2p</code>	<code>false</code>	No	Enables the use of the shoal P2P framework to distribute conference messages across the server pool.

Table 28–7 lists the properties for Shoal access across subnets.

Table 28–7 Properties for Shoal Access Across Subnets

Property	Default Value	Required	Description
<code>relay.imadmin.enable</code>	<code>true</code>	Yes	Starts the relay server.
<code>relay.listen_address= address of relay server</code>	None	Optional	Specifies the address of the relay server.
<code>relay.uri_list</code>	None	Yes	Displays the list of relay servers added.

Multiplexor Configuration Properties

Table 28–8 lists and describes the multiplexor configuration properties.

Table 28–8 Multiplexor Configuration Properties

Property	Default Value	Description
<code>iim_mux.listenport</code>	<i>multiplexorname</i> or <i>IP address:5222</i>	IP address or FQDN and listening port on which the multiplexor listens for incoming requests from clients. The value format is <i>IPaddress:port</i> or <i>multiplexorname:port</i> . If no IP address or domain name is listed, INADDR_ANY on localhost is assumed. If you change this value, also change the im.html and im.jnlp files so that they match the port value.
<code>iim_mux.serverport</code>	45222	The Instant Messaging Server instance and port the multiplexor communicates to. The value format is a comma-separated list of <i>host_name:port</i> .
<code>iim_server.usemuxport</code>	true	Enables the multiplexor listen port on the server.
<code>iim_mux.maxthreads</code>	5	Maximum number of threads per instance of the multiplexor.
<code>iim_mux.maxsessions</code>	5000	Maximum number of concurrent connections per multiplexor process.
<code>iim_mux.keydbprefix</code>	None	This value should contain the key database file name prefix. The key database file name must always end with key3.db . If the Key database contains a prefix, for example This-Database-key3.db , then value of this property is This-Database .

Table 28–8 (Cont.) Multiplexor Configuration Properties

Property	Default Value	Description
<code>iim_mux.certdbprefix</code>	None	This value should contain the certificate database file name prefix. The certificate database file name must always end with cert7.db . If the certificate database contains a prefix, for example Secret-stuff-cert7.db , then value of this property is Secret-stuff .
<code>iim_mux.certnickname</code>	<i>Multiplexor-Cert</i>	This value should contain the name of the certificate you entered while installing the certificate. The certificate name is case-sensitive.
<code>iim_mux.enable</code>	true	If the value is true , the multiplexor will run for this instance. If the value is false , the multiplexor will not run for this instance.
<code>iim_mux.log4j.refresh</code>	60	Specifies the number of seconds in which the multiplexor rereads its log4j configuration.
<code>iim_mux.threadpool.capacity</code>	-1	Specifies the queue size for the multiplexor's default thread pool.
<code>iim_mux.jvm.maxmemorysize</code>	256	The maximum heap size, in MB, that the JVM running the multiplexor is allowed to use. Used to construct the -mx argument of the Java command.
<code>iim_mux.polling_interval</code>	5	Specifies the polling interval, in seconds, during which the multiplexor attempts to contact the next Instant Messaging Server for failover. To disable polling, specify a negative value.
<code>iim_mux.monitor.enable</code>	false	Enables monitoring for the multiplexor when the value is true .
<code>iim_mux.monitor.refreshtimeout</code>	30	Specifies the common refresh timeout for monitoring statistics cache of multiplexor.
<code>iim_mux.admin.user</code>	None	Specifies the administrative user of the multiplexor
<code>iim_mux.admin.password</code>	None	Specifies the password for the multiplexor administrative user.

Archive Properties

[Table 28–9](#) lists the properties you use to manage Instant Messaging Server archiving.

Table 28–9 Archive Properties

Property	Default Value	Description
iim_arch.readacl.admin	None	Contains the administrator's DN. Multiple values should be separated by a semi colon (;).
iim_arch.readacl.adminonly	false	Contains true or false . true : Only the administrator's DN, as specified by the property iim_arch.readacl.admin is added to the ReadACL field, overwriting the default behavior of the ReadACL field. false : The administrator's DN, as specified by the property iim_arch.readacl.admin is added to the ReadACL field in addition to the default behavior.
iim_arch.categories	all	Contains a comma-separated list of message types that can be archived. The following types are available: <ul style="list-style-type: none"> ▪ poll ▪ alert ▪ chat ▪ conference ▪ news
iim_arch.categoryname	None	If a category name is not assigned for any of the categories, the value of this property is used as the category name.
iim_arch.alert.categoryname	None	Contains the name of the category containing the archived alert messages. There is no requirement to dedicate a category to alert messages.
iim_arch.poll.categoryname	None	Contains the name of the category containing the archived poll messages. There is no requirement to dedicate a category to poll messages.
iim_arch.conference.categoryname	None	Contains the name of the category containing the archived conference messages. There is no requirement to dedicate a category to conference messages.
iim_arch.chat.categoryname	Name	Contains the name of the category containing the archived chat messages. There is no requirement to dedicate a category to chat messages.
iim_arch.news.categoryname	None	Contains the name of the category containing the archived news messages. There is no requirement to dedicate a category to news messages.
iim_arch.conference.quiettime	5	Contains the maximum duration of silence between two consecutive messages in a room (both public and private) after which the RD expires and a new RD is created for archiving the message. The value is in minutes.
iim_arch.poll.maxwaittime	15	Contains the maximum time for which poll data is buffered in the server. The value is in minutes.
iim_arch.admin.email	None	A comma-separated list of administrator email addresses.
iim_arch.alert.admin.email	None	A comma-separated list of administrator email addresses to which all archived alert messages are sent. This property overrides iim_arch.admin.email for alert messages.
iim_arch.chat.admin.email	None	A comma-separated list of administrator email addresses to which all archived chat messages are sent. This property overrides iim_arch.admin.email for chat messages.

Table 28–9 (Cont.) Archive Properties

Property	Default Value	Description
<code>iim_arch.conference.admin.email</code>	None	A comma-separated list of administrator email addresses to which all archived conference messages are sent. This property overrides <code>iim_arch.admin.email</code> for conference messages.
<code>iim_arch.poll.admin.email</code>	None	A comma-separated list of administrator email addresses to which all archived poll messages are sent. This property overrides <code>iim_arch.admin.email</code> for poll messages.
<code>iim_arch.news.admin.email</code>	None	A comma-separated list of administrator email addresses to which all archived news messages are sent. This property overrides <code>iim_arch.admin.email</code> for news messages.
<code>iim_arch.email.archiveheader.name</code>	None	Name of the extended RFC 822 header.
<code>iim_arch.email.archiveheader.value</code>	all	Value corresponding to the header name for <code>iim_arch.email.archiveheader.name</code> .

Watchdog Properties

The watchdog monitors the server process and attempts to restart the server if it determines that the server is not running. See ["Managing the Watchdog Process"](#) for more information.

[Table 28–10](#) lists and describes the watchdog configuration properties.

Table 28–10 Watchdog Configuration Properties

Property	Default Value	Description
<code>iim_wd.enable</code>	true	Enables the watchdog feature. To reset this property or disable the watchdog, set this to false . To avoid conflicts, you should disable the watchdog if you are monitoring the Instant Messaging server using the operating system administration console.
<code>iim_wd.period</code>	300 (seconds)	The watchdog periodically polls the server to check whether it is running. This property sets the interval between two status polls.
<code>iim_wd.maxRetries</code>	3	Sets the number of retries, times the watchdog will attempt to contact the Instant Messaging server, before shutting down and restarting the server. The maximum is ten retries.
<code>iim_wd.log4j.refresh</code>	60	Specifies in seconds how often the watchdog rereads its log4j configuration.
<code>iim_wd.pidfile</code>	None	The file that stores the watch dog's PID.
<code>iim_wd.jvm.maxmemorysize</code>	256 (MB)	The maximum heap size, in MB, that the JVM running the watchdog is allowed to use. Used to construct the -mx argument of the Java command.
<code>iim_wd.debug_on_restart</code>	true	When a server is unresponsive, and if the property <code>iim_wd.debug_on_restart</code> is set to true, it captures a jstack dump of the server process in the log directory, when the server is hung.

Monitoring Properties

The properties in [Table 28–11](#) configure the way the server interacts with the Oracle Enterprise System Monitoring Framework.

Table 28–11 Monitoring Properties

Property	Default Value	Description
<code>iim_server.monitor.enable</code>	<code>false</code>	Used by the Oracle Enterprise System Monitoring Framework. If true , configures the server to make its activities available to mfwk . Otherwise, the server does not make its activities available.
<code>iim_server.monitor.htmlport</code>	None	If specified, opens the JMX HTML adaptor port on the specified port. By default, this port is not enabled as opening the port can present a security risk.

Agent Properties

Agents, such as the Calendar agent, enable functionality within Instant Messaging Server and enhance its interoperability with other Unified Communications Suite servers.

[Table 28–12](#) lists and describes agent configuration properties.

Table 28–12 Agent Configuration Properties

Property	Default Value	Description
<code>agent-calendar.broker.address</code>	None	Specifies the host name and port on which the broker is running.
<code>agent-calendar.broker.password</code>	None	Specifies the password of the broker user name.
<code>agent-calendar.broker.user</code>	None	Specifies the broker user name.
<code>agent-calendar.consumer.params</code>	None	Specifies any extra properties required by the calendar notifications consumer.
<code>agent-calendar.consumer.topic</code>	None	Topic on which the calendar notifications are delivered.
<code>agent-calendar.iim_server.host</code>	<code>localhost</code>	Host name of the Instant Messaging server with which the agent calendar communicates.
<code>agent-calendar.iim_server.port</code>	<code>\$iim_server.port</code>	Port number of the Instant Messaging server with which the agent calendar communicates.
<code>agent-calendar.imadmin.enable</code>	<code>false</code>	If set to true , you can start the agent-calendar by using the imadmin command.
<code>agent-calendar.log4j.refresh</code>	<code>60</code>	Specifies in seconds how often the calendar agent rereads the log4j configuration.
<code>agent-calendar.serveralarms.contentType</code>	<code>text/calendar</code>	Specifies the content type used to deliver calendar alerts. Can be text/plain or text/ical .
<code>iim_agent.enable</code>	<code>false</code>	If TRUE , iim.conf , enables Instant Messaging Server agents. Set the value to FALSE , or remove the property from iim.conf to disable all agents.
<code>iim_agent.agent-calendar.enable</code>	<code>false</code>	Used with the Calendar agent. If TRUE or absent from iim.conf , loads a component that enables the Calendar agent specifically.
<code>agent-calendar.jid</code>	None	The JID of the Calendar agent.
<code>agent-calendar.password</code>	None	Defines the password with which the Calendar agent connects to the Instant Messaging server.
<code>agent-calendar.imadmin.enable</code>	<code>false</code>	Start the agent-calendar by using the imadmin command if set to true .

JMQ Properties

Table 28–13 lists the calendar agent properties.

Table 28–13 JMQ Properties

Property	Default Value	Description
agent-calendar.broker.address	None	Specifies the host name and port on which the broker is running.
agent-calendar.broker.password	None	Specifies the password of the broker user name.
agent-calendar.broker.user	None	Specifies the broker user name.
agent-calendar.consumer.params	None	Specifies any extra properties required by the calendar notifications consumer.
agent-calendar.consumer.topic	None	Topic on which the calendar notifications are delivered.
agent-calendar.iim_server.host	localhost	Host name of the Instant Messaging server with which the agent calendar communicates.
agent-calendar.iim_server.port	<i>\$iim_server.port</i>	Port number of the Instant Messaging server with which the agent calendar communicates.
agent-calendar.imadmin.enable	false	Start the agent-calendar by using the imadmin command if set to true .
agent-calendar.log4j.refresh	60	Specifies in seconds how often the calendar agent rereads the log4j configuration.
agent-calendar.serveralarms.contentType	text/calendar	Specifies the content type used to deliver calendar alerts. Can be text/plain or text/ical .
iim_agent.enable	false	If TRUE , iim.conf , enables Instant Messaging Server agents. Set the value to FALSE , or remove the property from iim.conf to disable all agents.
iim_agent.agent-calendar.enable	None	Used with the Calendar agent. If TRUE or absent from iim.conf , loads a component that enables the Calendar agent specifically.
agent-calendar.jid	None	The JID of the Calendar agent.
agent-calendar.password	None	Defines the password with which the Calendar agent connects to the Instant Messaging server.
agent-calendar.imadmin.enable	None	Start the agent-calendar by using the imadmin command, if set to true .

HTTP/XMPP Gateway Properties

Table 28–14 lists the properties you use to bind to the HTTP/XMPP gateway.

Table 28–14 HTTP/XMPP Gateway Properties

Property	Default Value	Description
httpbind.jid	httpbind.\${iim_server.domainname}	A JID to bind the HTTP/XMPP gateway.
httpbind.password	random	Password to authenticate the HTTP/XMPP gateway to the Instant Messaging server.

SMS Integration Properties

Table 28–15 lists the SMS integration properties.

Table 28–15 SMS Integration Properties

Property	Default Value	Description
<code>msgw.imadmin.enable</code>	false	Enables or disables the SMS gateway. If set to true , you can start the SMS gateway by using the imadmin command.
<code>msgw.jid</code>	None	A JID to bind the SMS gateway to the Instant Messaging server. The value of this property should be the same as the value that you define for the smppbind.jid property.
<code>msgw.password</code>	<i>random</i>	Password to authenticate the SMS gateway to the Instant Messaging server. The value of this property should be the same as the value that you define for the smppbind.password property.
<code>msgw.iim_server</code>	None	Host name and port number of the Instant Messaging server.
<code>msgw.sms_limit</code>	-1	Number of messages that can be sent per hour. The default value is -1 and it indicates that unlimited number of SMS messages that can be sent per hour.
<code>msgw.sms_queue_capacity</code>	512	Maximum number of messages that can be queued for SMS delivery.
<code>msgw.im_char_limit</code>	500	Maximum number of characters that you can specify in one message. If the number of characters is greater than the specified value, the message is rejected.
<code>smpp.smsc_ip_address</code>	None	IP address or host name of the SMSC.
<code>smpp.smsc_port</code>	2775	Port number of the SMSC.
<code>smpp.bind_id</code>	None	Identifier used to bind the SMS gateway to the SMSC.
<code>smpp.bind_password</code>	<i>random</i>	Password to authenticate the SMS gateway to the SMSC.
<code>smpp.sender_id</code>	None	Sender ID of the outgoing SMS.
<code>iim_agent.smppbind.enable</code>	false	Enables the Instant Messaging server to identify the SMS gateway.
<code>smppbind.jid</code>	None	A JID for binding the SMS gateway to the Instant Messaging server.
<code>smppbind.password</code>	<i>random</i>	Password to authenticate the SMS gateway to the Instant Messaging server.
<code>smpp.bind_usessl</code>	false	Specifies whether to use SSL when connecting to the SMSC.
<code>smppbind.log4j.refresh</code>	60	Specifies in seconds how often the SMS gateway rereads its log4j configuration.
<code>msgw.im_char_limit</code>	500	The maximum number of characters permitted in an SMS message.

XMPP WebSocket Gateway Properties

Table 28–16 lists the XMPP WebSocket Gateway properties

Table 28–16 XMPP WebSocket Gateway Properties

Property	Default Value	Description
<code>iim_mux.encrypted_channel</code>	false	When a channel is created, you can use this property to specify it as encrypted.
<code>iim_mux.maxthreads</code>	10	Maximum number of threads per instance of the multiplexor.
<code>iim_mux.maxsessions</code>	5000	Maximum number of concurrent connections per multiplexor process.
<code>iim_mux.monitor.enable</code>	false	Enables monitoring for the multiplexor when the value is true .

Table 28–16 (Cont.) XMPP WebSocket Gateway Properties

Property	Default Value	Description
<code>iim_mux.monitor.refreshtimeout</code>	30	Specifies the common refresh timeout for monitoring statistics cache of multiplexor.
<code>iim_mux.polling_interval</code>	5	Specifies the polling interval, in seconds, during which the multiplexor attempts to contact the next Instant Messaging Server for failover. To disable polling, specify a negative value.
<code>iim_websocket.session_idletimeout</code>	-1	Specifying 0 or a negative number disables timeout and the session will never close due to inactivity. Specifying a positive non-zero number in seconds results in the session closing after the specified time if idle. This property is optional.
<code>iim.log4j.config</code>	None	Specifies the location of the log4j.conf file.
<code>iim_mux.serverport</code>	45222	The Instant Messaging Server instance and port the multiplexor communicates to. The value format is a comma-separated list of <i>host_name:port</i>
<code>iim_mux.usessl</code>	false	Specifies whether the multiplexor uses SSL. The usage of SSL is currently limited to encrypted data between the multiplexor and server.
<code>iim_mux.sslkeystore</code>	None	Specifies the keystore file for multiplexor to server encryption. This property is optional.
<code>iim_mux.keystorepasswordfile</code>	None	Specifies the keystore password file for multiplexor to server encryption. This property is optional.
<code>iim_mux.trust_all_cert</code>	false	Determines whether the multiplexor trusts all certificates that are presented, even if a certificate fails verification.

Instant Messaging Server APIs

This chapter describes the APIs used by Oracle Communications Instant Messaging Server.

Instant Messaging Server APIs Overview

Instant Messaging Server provides Java APIs that can be used to develop extension or integration modules. Detailed documentation of these APIs is provided in the form of HTML files generated by Javadocs. The Javadoc files are installed in the *InstantMessaging_home/html/apidoc* directory.

The following Instant Messaging Server APIs are available:

- [Instant Messaging Server Service API](#)
- [Service Provider Interfaces](#)

Instant Messaging Server Service API

The Instant Messaging Server Service API is used by the applications located on the same host or in the remote host to access Instant Messaging Server services, such as presence and conferences.

The Instant Messaging Server Service API can be used for:

- A Java-based or web-based client
- A bridge or a gateway to enable another class of clients
- Integration of presence into existing applications

Service Provider Interfaces

The Service Provider Interface APIs provide the ability to extend Instant Messaging Server functionality. The Service Provider Interface is composed of the following independent APIs:

- [Archive Provider API](#)
- [Message Conversion API](#)
- [Authentication Provider API](#)

Archive Provider API

An Archive Provider is a software module usually providing integration with the archive or auditing system. Each configured Archive Provider is invoked for each server process.

The Archive Provider is invoked for the following server processes:

- When an instant message is sent
- During an authentication event, such as login or logout
- When there is a change in presence status
- During a subscription event, for example, when someone joins or leaves a conference

Examples of applications that can use the Archive Provider API are:

- Instant Messaging Server Archive
The default Instant Messaging Server archive in Instant Messaging Server is based on the Archive Provider API. See ["Managing Archiving in Instant Messaging Server"](#) for more information on archiving.
- The application that records the usage statistics for sizing purposes.

Message Conversion API

A Message Converter is invoked for every message or each message part going through the server. The Message Converter may leave the message part intact or modify or remove the message part. The text parts are processed as Java String Objects. The Message Converter processes other attachments as a stream of bytes and returns a potentially different stream of bytes, or nothing at all if the attachment is to be removed.

Examples of applications that can use the Message Conversion API are as follows:

- Virus checking and removal
- Translation engine integration
- Message content filtering

See ["Managing Message Conversion in Instant Messaging"](#) for more information on converting messages.

Authentication Provider API

The Authentication Provider API provides the ability to deploy Instant Messaging Server in environments that are not using Access Manager password-based or token-based authentication service. This API is invoked whenever an end user requests authentication, and it can be used with the LDAP authentication.

Web Presence API

The Web Presence API makes it possible to retrieve presence information about users connected to the Instant Messaging Server. An example of an application that might use the API would be an enterprise application that provides a presence widget that allows employees to see the presence status of other employees.

See ["Using the Web Presence API"](#) for more information about the Web Presence API.

imadmin Command Reference

This chapter describes how to use the **imadmin** command to administer Oracle Communications Instant Messaging Server.

imadmin Overview

You can use the **imadmin** command to start, stop, and refresh the Instant Messaging server and multiplexor.

imadmin Requirements

You must invoke the **imadmin** command from the host on which Instant Messaging server is installed. Run **imadmin** as root or as the end user you specified during configuration.

imadmin Location

By default, **imadmin** is installed in the following location:

InstantMessaging_home/bin

imadmin Commands

[Table 30–1](#) lists and describes commands related to the **imadmin** command.

Table 30–1 imadmin Commands and Descriptions

Command	Description
imadmin assign_services	<p>If iim.policy.modules is set to iim_ldap, and iim.userprops.store is set to ldap, this command adds object classes (sunIMUser, and sunPresenceUser) to user entries in the directory. Instant Messaging Server requires these object classes to store properties in LDAP. The assign_services command fails if the LDAP search limit exceeds the defined value. To avoid this failure, increase the search limit of the LDAP server. Set the values of the directory server parameters as follows:</p> <ul style="list-style-type: none"> To set unlimited limit for the search size, type: dsconf set-server-prop search-size-limit: unlimited To set unlimited limit for the search time, type: dsconf set-server-prop search-time-limit: unlimited

Table 30–1 (Cont.) imadmin Commands and Descriptions

Command	Description
imadmin status	Checks to see if the components (server, multiplexor, agent-calendar, and watchdog) are up and running and displays the results. If you do not specify a component, the imadmin command returns information about all components.
imadmin start	Starts the enabled component(s).
imadmin stop	Stops the enabled component(s).
imadmin refresh	Refreshes the enabled component(s).
imadmin start server	Starts only the server.
imadmin stop server	Stops only the server.
imadmin refresh server	Refreshes only the server.
imadmin start multiplexor	Starts only the multiplexor.
imadmin stop multiplexor	Stops only the multiplexor.
imadmin refresh multiplexor	Refreshes only the multiplexor.
imadmin start agent-calendar	Starts only the Calendar agent.
imadmin stop agent-calendar	Stops only the Calendar agent.
imadmin refresh agent-calendar	Refreshes only the Calendar agent.
imadmin start watchdog	Starts only the watchdog.
imadmin stop watchdog	Stops only the watchdog.
imadmin refresh watchdog	Refreshes only the watchdog.
imadmin version	Displays the version.
imadmin start sms-gateway	Starts the SMS Gateway.
imadmin stop sms-gateway	Stops the SMS Gateway.
imadmin status sms-gateway	Displays the status of SMS Gateway.
imadmin refresh sms-gateway	Refreshes the SMS Gateway.
imadmin start relay	Starts the relay server.
imadmin stop relay	Stops the relay server.
imadmin refresh relay	Refreshes the relay server.
imadmin status relay	Displays the status of relay server.
imadmin smf-register	Registers the Service Management Facility (SMF) with Instant Messaging Server.

imadmin Syntax

```
imadmin [ options ] [ action ] [ component ]
```

imadmin Options

Table 30–2 lists and describes options for the **imadmin** command.

Table 30–2 Options for imadmin Command

Option	Description
-c <i>alt-config-file</i>	Used with the start and refresh actions, to specify a configuration file other than <code>/etc/opt/sun/comms/im/default/config/iim.conf.xml</code> file.
-h	Displays help on the imadmin command.

imadmin Actions

Table 30–3 lists and describes actions performed after various **imadmin** commands are issued.

Table 30–3 Actions for imadmin Command

Action	Description
status	Returns information about Instant Messaging Server components (server, multiplexor, agent-calendar, and watchdog). You do not need to provide a <i>component</i> with this action.
start	Sets the classpath , the Java heap size and starts all the specified components.
stop	Stops all the specified component's daemons.
refresh	Stops and starts the specified component(s). Useful after a configuration change.

imadmin Components

Table 30–4 lists and describes the components for the **imadmin** command.

Table 30–4 Components for imadmin Command

Components	Description
agent-calendar	Indicates the Calendar agent (agent-calendar).
multiplexor	Indicates the multiplexor alone.
server	Indicates the Instant Messaging server.
watchdog	Indicates the watchdog.
sms-gateway	Indicates the SMS Gateway.
relay	Indicates shoal cross-subnet relay server.

imconfutil Command Reference

The **imconfutil** command enables you to set, modify, and list Oracle Communications Instant Messaging Server configuration properties. This chapter describes the **imconfutil** command. It gives the tool's syntax, commands and command options, and examples of its usage.

Requirements: Must be run locally as **root** on the Instant Messaging Server host.

Location on UNIX: *InstantMessaging_home/sbin*

Syntax

The syntax of the **imconfutil** command is:

```
imconfutil [-c config-file] [--quiet] command [command-specific options] --help
```

Options

Table 31–1 shows the options for the **imconfutil** command.

Table 31–1 *imconfutil Options*

Option	Description
-c <i>configuration_file</i>	Specifies the iim.conf.xml file, for example, /opt/sun/comms/im/config/iim.conf.xml .
-h --help	Prints help information.
-q --quiet	The output of the command prints only the value, not the property name.
<i>command</i> [<i>command-specific options</i>]	Administers configuration properties and descriptions.

Table 31–2 shows **imconfutil** commands and command-specific options. Options that are, in fact, optional appear in brackets.

Table 31–2 imconfutil Commands and Command-specific Options

Command and Command-specific Options	Description
set-prop <i>property</i>	Sets a configuration property.
del-prop <i>property</i>	Deletes a configuration property.
get-prop <i>property</i>	Lists the value of a configuration property. The get-prop command, with the -c option, but without a specific property, displays the values of all properties.
verify <i>property password</i>	Verifies that a specified password property (for example, iim_ldap.usergroupbindcred) is set to the given cleartext string in the iim.conf.xml file.
add-ldap-replica id=id host=host [port=port] [usessl=true false]	Adds a new LDAP replica description. <i>id</i> is a unique name for the LDAP replica. <i>host</i> is the host name or IP address of the LDAP replica. <i>port</i> is the port number on which the LDAP replica listens. usessl specifies whether to use SSL when connecting to this LDAP replica.
delete-ldap-replica <i>id</i>	Deletes an existing LDAP replica description entry.
set-ldap-replica-prop <i>id</i> [host=host] [port=port] [usessl= true false]	Modifies the properties of an LDAP replica description entry. <i>id</i> is a unique name for the LDAP replica. <i>host</i> is the host name or IP address of the LDAP replica. <i>port</i> is the port number on which the LDAP replica listens. usessl specifies whether to use SSL when connecting to this LDAP replica.
get-ldap-replica-prop <i>id</i>	Lists the properties of an LDAP replica description.
list-ldap-replicas	Displays a list of LDAP replica description entries.
add-component id=id jid=jid [password=password] [broadcastpresence=true false]	Adds a component. <i>id</i> can be any string, but it is convenient to use something that identifies the component, such as calagent , httpbind , msggateway , aimgateway , msggateway , or yimgateway . <i>jid</i> is the jabber ID to bind the component to the Instant Messaging server. <i>password</i> authenticates the component to the Instant Messaging server. broadcastpresence is for use when the Web Presence API is added. If set to true, it allows Instant Messaging Server to send presence information to the Web Presence API.
delete-component <i>id</i>	Deletes an existing component description entry.
set-component-prop <i>id</i> <i>property= value</i> [<i>property= value...</i>]	Modifies the properties of a component description entry.
get-component-prop <i>id</i>	Lists the properties of a component description.
verify-component-pass <i>id password</i>	Verifies that the component with the specified ID has the specified cleartext password in the iim.conf.xml file.
list-components	Displays a list of component description entries.
add-coserver <i>id serverid password host</i> [requiressl= true false] <i>domain</i>	Adds a new coserver description. <i>id</i> is a unique name for this coserver entry. <i>serverid</i> is the server ID used to identify this coserver, and should be the same as mentioned in that coserver's configuration. <i>password</i> is used to authenticate this coserver. <i>host</i> is the host name or IP address of this coserver. requiressl specifies whether TLS is required when connecting to this coserver. <i>domains</i> specifies the domain served by this coserver.
delete-coserver <i>id</i>	Deletes an existing coserver description entry.
set-coserver-prop <i>id</i> <i>property= value</i> [<i>property= value...</i>]	Modifies the properties of a coserver description entry.
get-coserver-prop <i>id</i> [<i>server id password host</i>] [requiressl=true false domain]	Shows the properties of a coserver description. <i>id</i> is a unique name for this coserver entry. <i>serverid</i> is the server ID used to identify this coserver, and should be the same as mentioned in that coserver's configuration. <i>password</i> is used to authenticate this coserver. <i>host</i> is the host name or IP address of this coserver. requiressl specifies whether TLS is required when connecting to this coserver. <i>domains</i> specifies the domain served by this coserver.

Table 31–2 (Cont.) imconfutil Commands and Command-specific Options

Command and Command-specific Options	Description
verify-coserver-pass <i>id password</i>	Verifies that the coserver whose ID is <i>id</i> has the specified cleartext password in the iim.conf.xml file.
list-coservers	Displays a list of coserver description entries.
add-server-threadpool <i>id=id</i> <i>maxthreads=maxthreads</i> <i>[capacity=capacity]</i>	Adds a new server threadpool entry.
delete-server-threadpool	Deletes an existing server threadpool entry.
set-server-threadpool-prop	Modifies the properties of a server threadpool entry.
get-server-threadpool readpool-prop <i>id</i> <i>maxthreads=maxthreads</i> <i>capacity=capacity</i>	Displays the properties of a server threadpool entry.
list-server-threadpools	Displays a list of server threadpools.
set-default-server-threadpool <i>id</i>	Sets the default server threadpool.
get-default-server-threadpool	Displays the default server threadpool ID.
add-mux-threadpool <i>id=id</i> <i>maxthreads=maxthreads</i> <i>[capacity=capacity]</i>	Adds a new multiplexor threadpool entry.
delete-mux-threadpool <i>id</i>	Deletes an existing multiplexor threadpool entry.
set-mux-threadpool-prop <i>id</i> <i>property= value [property= value...]</i>	Modifies the properties of a multiplexor threadpool entry.
get-mux-threadpool-prop <i>id</i> <i>maxthreads capacity</i>	Displays the properties of a multiplexor threadpool entry.
list-mux-threadpools	Displays a list of multiplexor threadpools.
set-default-mux-threadpool <i>id</i>	Sets the default multiplexor threadpool.
get-default-mux-threadpool	Displays the default multiplexor threadpool ID.
add-listener <i>id=id port=port</i> <i>protocols=protocols</i> <i>[property=value...]</i>	Adds a new listener entry.
delete-listener <i>id</i>	Deletes a listener entry.
set-listener-prop <i>id property= value</i> <i>[property= value...]</i>	Modifies the properties of a listener entry.
get-listener-prop <i>id [property1</i> <i>property2...]</i>	Displays the properties of a listener entry.
list-listeners	Displays a list of listeners.
add-jmqbrokerid= <i>id address=</i> <i>hostnameport: user=broker_name</i> <i>password=broker_password</i>	Adds a JMQ broker for handling calendar availability notifications.
delete-jmqbroker <i>id</i>	Deletes a JMQ broker.
list-jmqbrokers	Lists the IDs of all JMQ brokers. The properties of an individual broker can then be obtained using the imconfutil get-jmqbroker-prop command.

Table 31–2 (Cont.) imconfutil Commands and Command-specific Options

Command and Command-specific Options	Description
set-jmqbroker-prop <i>id</i> <i>property=value</i> [<i>property=value...</i>]	Sets one or more properties of a JMQ broker. To set more than one property, enter a space-separated list of <i>property=value</i> pairs. The properties you can set are: <ul style="list-style-type: none"> ■ address: the host name and port (<i>hostname:port</i>) of the JMQ publisher that the calendar-agent broker communicates with. The JMQ publisher is a part of Calendar Server. ■ user: The user name for the broker to use in connecting to Instant Messaging Server. ■ password: The password for the broker to use in connecting to Instant Messaging Server.
get-jmqbroker-prop <i>id</i>	Gets the properties of the JMQ broker with the specified ID.
verify-jmqbroker-pass <i>id</i> <i>password</i>	Verifies that the JMQ broker whose ID is <i>id</i> has the specified cleartext password in the iim.conf.xml file.
generate-password [-k keyfile <i>path_to_key_file</i>] <i>plaintext_password</i>	Encrypts a given <i>plaintext_password</i> and prints it out. Use this command when to change an individual password in an Instant Messaging Server state file. For information on state files, see <i>Instant Messaging Server Installation and Configuration Guide</i> . If you do not provide the <i>plaintext_password</i> with the generate-password command, the imconfig utility prompts you to enter the <i>plaintext_password</i> twice, and then generates the encrypted password. The -k option is mandatory.
rekey	Generate a new encryption key and newly encrypted passwords for all current Instant Messaging Server passwords.
generate-key	Generate an encryption key and output it to the console. You can send the key to a file, which can then be used to create new encrypted passwords for a silent configuration.
add-plugin <i>id</i> [enable] [<i>handlerclass</i>] [<i>namespace</i>] <i>type</i>	Adds a new plugin.
delete-plugin <i>id</i>	Deletes a plugin.
set-plugin-prop <i>id</i> <i>property=value</i> [<i>property=value ...</i>]	Modifies plugin properties.
get-plugin-prop <i>id</i> [enable] [<i>handlerclass</i>] [<i>namespace</i>] [<i>type</i>]	Shows plugin properties.
list-plugins	Shows all plugins.

Example imconfutil Commands

The following are examples of using the **imconfutil** command.

- To print a list of command options:

```
imconfutil
```
- To use the specified configuration file and print the value of the **iim.instancedir** property:

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml get-prop iim.instancedir
```
- To get help for a command:

```
imconfutil set-prop --help
```

- To set a single property:

```
imconfutil set-prop -c /opt/sun/comms/im/config/iim.conf.xml
iim.instancedir=/var/opt/SUNWiim/default
```

- To set multiple properties:

```
imconfutil set-prop -c /opt/sun/comms/im/config/iim.conf.xml
iim.instancevardir=/var/opt/SUNWiim/default
iim.instancedir=/etc/opt/SUNWiim/default
```

- To add a component:

```
imconfutil add-component -c /opt/sun/comms/im/config/iim.conf.xml calagent
jid=calendar.example.com password=password
```

- To print the value of all enabled (configured) components:

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml list-components
```

- To display all properties currently set in the **iim.conf.xml** file:

```
imconfutil -c /opt/sun/comms/im/iim.conf.xml get-prop
```

- To delete a component:

```
imconfutil delete-component -c /opt/sun/comms/im/config/iim.conf.xml calagent
```

- To delete a property:

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml del-prop iim_
ldap.resynctime
```

- To display a component's property:

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml get-component-prop calagent
```

- To add an LDAP replica:

```
imconfutil add-ldap-replica -c /opt/sun/comms/im/config/iim.conf.xml id=ldap1
hostname=im.example.com port=1389 usessl=false
```

- To print both the restricted and unrestricted list of commands and options (-u switch):

```
imconfutil -u
```

- To set a restricted property:

```
imconfutil set-prop -u -c /opt/sun/comms/im/config/iim.conf.xml msgw.sms_
limit=-1
```

- To verify a password property:

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml verify iim_
ldap.usergroupbindcred admin
```

- To verify a component password:

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml verify-component-pass
httpbind admin
```

- To verify a cosever password:

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml verify-coserver-pass  
coserver1 admin
```

- To add a JMQ broker:

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml add-jmqbroker id=broker1  
address=jmqbroker.example.com:7676 user=thisjmqbroker password=zyxw
```

- To generate an encrypted password for replacing a password in a state file:

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml generate-password -k drowp
```

- To generate a new encryption key and newly encrypted passwords for all current Instant Messaging Server passwords:

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml rekey
```

- To generate a key and send it to a file:

```
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml generate-key >  
/tmp/mykey.txt  
imconfutil -c /opt/sun/comms/im/config/iim.conf.xml generate-password -k  
/tmp/mykey.txt plaintext_password
```

iwadmin Command Reference

This chapter describes the **iwadmin** command. It gives the tool's syntax, commands and command options, and examples of its usage.

Overview of the iwadmin Command

The **iwadmin** command enables you to deploy, redeploy, and undeploy XMPP Web components that are installed with Oracle Communications Instant Messaging Server. See ["iwadmin Commands and Command-Specific Options"](#) for a list of these IM Web components. You can also use the **iwadmin** command to create a WAR file or a ZIP file that contains a WAR file and additional files needed for an XMPP Web component.

The location of the **iwadmin** command is: *InstantMessaging_home/sbin/*. You have to run **iwadmin** locally, as **root**, on the Instant Messaging Server host.

Syntax

The syntax of the **iwadmin** command is:

```
iwadmin [ iwadmin options ] command [ command-specific options ]
```

iwadmin Options

There are two **iwadmin** options. The options can be placed anywhere after the **iwadmin** command:

[Table 32–1](#) lists the **iwadmin** options that are available.

Table 32–1 *iwadmin Options*

Option	Description
[--verbose -v]	Enables verbose output.
[--force -f]	If the all command option is used (see "iwadmin Commands and Command-Specific Options") and the command fails on one Web application, this option forces a continuation to the next Web application.

iwadmin Commands and Command-Specific Options

Each of the **iwadmin** commands, with the exception of the **iwadmin list** command, requires specification of an option that corresponds to a Web component provided with Instant Messaging Server. [Table 32–2](#) lists the IM Web components.

Table 32–2 Options for Web Components Provided with the Instant Messaging Server

Web Component	Description
httpbind	Can be used with all iwadmin commands, except for the iwadmin list command.
presenceapi	This option corresponds to the Web Presence API. It can only be used with the iwadmin generatezip command. See "Using the Web Presence API" for more information on the Web Presence API

[Table 32–3](#) lists **iwadmin** commands and command-specific options.

Table 32–3 iwadmin Command and Command-Specific Options

Command and Command-Specific Options	Description
generatezip <i>IM_Web_component</i> [<i>--configfile</i> <i>-c</i>] [<i>--destination</i> <i>-d</i>]	<p>Creates a ZIP file for deploying Instant Messaging Server.</p> <p>The ZIP file contains:</p> <ul style="list-style-type: none"> ▪ A deployable WAR file for the specified IM Web component, either presenceapi or httpbind. ▪ A template file, either presenceapi.conf.template or httpbind.conf.template, to use for creating a configuration file for the specified IM Web component. ▪ A template file, either presenceapi_log4j.conf.template or httpbind_log4j.conf.template, for configuring a log4j log file for the specified IM Web component. ▪ The Instant Messaging Server password tool. See "passwordtool Command Reference" for more information. <p>The --configfile or -c parameter specifies the future location of the configuration file for the Web Presence API, once the ZIP file is unzipped. The location of the configuration file must be given as an absolute path.</p> <p>The --destination or -d parameter specifies the destination directory and file name for the ZIP file. The destination must be given as an absolute path and the destination directory must exist before you generate the ZIP file.</p>

Example iwadmin Commands

The following examples show how to use the **iwadmin** command.

- To print a summary of **iwadmin** usage:

```
iwadmin
```
- To create a ZIP file containing a WAR file and other files needed for deploying the **presenceapi** IM Web component (see **generatezip** in [Table 32–3](#)):

```
iwadmin generatezip presenceapi -c /opt/components/presence/config -d
/opt/components/presence/config/deploy/presenceapi.zip
```

passwordtool Command Reference

This chapter describes the Oracle Communications Instant Messaging Server **passwordtool** command.

passwordtool Overview

The **passwordtool** command makes it possible to:

- Generate encryption keys for passwords
- Regenerate encryption keys for passwords
- Generate encrypted passwords
- Verify that an unencrypted password matches an encrypted password

In versions prior to Instant Messaging Server 9.0.1.4.0, you had to enter unencrypted passwords in configuration files.

You can use the password tool with the following Web components that are provided with Instant Messaging Server:

- HTTPBIND
- Web Presence API

See ["Examples"](#) for how to use the password tool.

Requirements: Must be run locally as **root**.

Location: *InstantMessaging_home/sbin/passwordtool*

Syntax

The syntax of the **passwordtool** command is:

```
passwordtool [ IM_Web_component ] [ command ] [ command specific options ]
```

where *IM_Web_component* is one of the following:

- **httpbind**
- **presenceapi**

[Table 33–1](#) shows **passwordtool** commands and command-specific options.

Table 33–1 passwordtool Commands and Command-Specific Options

Command and Command-Specific Options	Description
generate-key	Generates a key for encrypting passwords. Passwords are generated using the generate command.
rekey	Creates a new key for generating passwords. Use this command when you think the current key may be compromised or when it has been in use for a long time and you want to change it. When you create a new key, create a new password to replace the password generated from the old key.
generate <i>cleartext-password</i>	Encrypts a given <i>cleartext-password</i> . You can then enter the encrypted password in the configuration file for an IM Web component. Before generating a password, you must use the generate-key or rekey command to generate a key for the password.
verify <i>plaintext-password</i> <i>encrypted-password</i>	Given a <i>plaintext-password</i> and an <i>encrypted-password</i> , verifies that the two match.

Examples

The following examples show how to use the Instant Messaging Server **passwordtool** command. You must be logged in as **root**.

To generate a key for creating passwords, change to the directory that contains the configuration file for the IM Web component and use the **generate-key** option:

```
cd config-file-dir
InstantMessaging_home/sbin/passwordtool IM_Web_component generate-key
```

To generate an encrypted password:

```
passwordtool IM_Web_component generate cleartext_password
```

Once generated, enter the password in the configuration file for the Web component.

To generate an encrypted password from the cleartext password *abcd* for the HTTPBIND component:

```
cd httpbind_config_dir
.../passwordtool httpbind generate-key
.../passwordtool httpbind generate abcd
MmHRfLCIB0ej5KGDqLC45Q==
```

The last line is the generated password.

To verify that a cleartext password and an encrypted password match:

```
.../passwordtool presenceapi verify drowp en!24Fedk#$dv==
```

To print a list showing **passwordtool** commands and command options:

```
passwordtool
Usage : passwordtool component{httpbind/presenceapi/sipgateway} generate password
passwordtool component{httpbind/presenceapi/sipgateway} verify cleartext-string
encrypted-string
passwordtool component{httpbind/presenceapi/sipgateway} generate-key
passwordtool component{httpbind/presenceapi/sipgateway} rekey
```

The **sipgateway** is currently not available for use with the password tool.

XMPP and HTTP Gateway Configuration Parameters

This chapter covers the Oracle Communications Instant Messaging Server configuration parameters used in the **httpbind.conf** configuration file.

Any time you modify the **httpbind.conf** file, you must restart the XMPP/HTTP Gateway by using the tools provided by your web container.

httpbind.conf File Location

By default, the **configure** utility creates the **httpbind.conf** file within the configuration directory (*InstantMessaging_cfg*) of the default server instance, for example:

- Solaris OS:
`/etc/opt/sun/comms/im/default/config/httpbind.conf`
- Red Hat Linux and Oracle Linux:
`/etc/opt/sun/im/default/config/httpbind.conf`

If you create multiple instances of Instant Messaging Server, the name of the **/default** directory varies depending on the instance. See the topic on creating multiple instances from a single Instant Messaging Server installation in *Instant Messaging Server Installation and Configuration Guide*. This file is created by the **configure** utility only in the default instance's *InstantMessaging_cfg* directory.

httpbind.conf File Syntax

The **httpbind.conf** file is a plain ASCII text file, with each line defining a gateway parameter and its value(s):

- A parameter and its value(s) are separated by an equal sign (=) with spaces and tabs allowed before or after the equal sign.
- A value can be enclosed in double quotes (" "). If a parameter allows multiple values, the entire value string must be enclosed in double quotes.
- A comment line must have an exclamation point (!) as the first character of the line. Comment lines are for informational purposes and are ignored by the server.
- If a parameter appears more than once, the value of the last parameter listed overrides the previous value.
- A backslash (\) is used for continuation and indicates the value(s) are longer than one line.
- Each line is terminated by a line terminator (\n, \r, or \r\n).

- The key consists of all the characters in the line starting with the first non-whitespace character and up to the first ASCII equal sign (=) or semi-colon (;). If the key is terminated by a semi-colon, it is followed by **lang-** and a tag that indicates the language in which this value is to be interpreted. The language tag is followed by an equal sign (=). All whitespace characters before and after the equal sign are ignored. All remaining characters on the line become part of the associated value string.
- Multiple values in the value string are separated using commas (,).
- Within a value, if any special characters like comma, space, newline, tab, double quotes, or backslash are present, the entire value needs to be within double quotes. In addition, every carriage return, line feed, tab, backslash, and double quotes within the value must be specified with a backslash (\).
- If you make changes to the **httpbind.conf** file, you must refresh the gateway's web container in order for the new configuration settings to take effect.

Note: The **httpbind.conf** file is initialized by the **configure** utility and should be modified only as described in this information.

How Load Balancing Occurs

HTTPBIND performs round-robin load balancing among the component sessions (connections from HTTPBIND to a back end) in a circular linked-list fashion to decide which back end is used.

A change in connection status is reflected almost immediately (as soon as a **StreamStatusChanged** event occurs). Thus, if a disconnection happens for a particular back end, it is out of the list. When the connection resumes, it comes back to the available list.

Instant Messaging Server XMPP/HTTP Gateway Configuration Parameters

[Table 34–1](#) describes the configuration parameters in **httpbind.conf**.

Table 34–1 XMPP/HTTP Gateway Configuration Parameters in `httpbind.conf`

Parameter	Default Value	Description
<code>httpbind.pool.nodeId</code>	Not applicable	If <code>httpbind.pool.support</code> is set to true , this parameter specifies the full URL for the server node in the server pool. This URL should not point to a load balancer, but to an Instant Messaging Server instance.
<code>httpbind.pool.support</code>	false	This parameter defines whether the gateway is in a server pool deployment. If no <code>httpbind.pool.nodeId</code> is specified, the value for this parameter is set to false . The value for this parameter can be: <ul style="list-style-type: none"> ■ true: The gateway is part of a server pool deployment. If you set this parameter to true, you must provide a value for <code>httpbind.pool.nodeId</code>. (enable, on, yes, and 1 are also valid values for this parameter.) ■ false (the default): The gateway is not part of a server pool deployment. Leaving the value blank (empty string) is also a valid value.
<code>httpbind.config</code>	Not applicable	Contains a comma-separated list of ID keys, or gwdomain-ids, which the gateway uses as a configuration key to determine which domains, hosts, host passwords, and component JIDs the gateway should use. See " httpbind.config ID Keys " for more information on ID keys.
<code>httpbind.content_type</code>	text/xml; charset=utf-8	The default value for the content-type HTTP header the gateway uses when sending a response back to the client.
<code>httpbind.hold</code>	Not applicable	Specifies the maximum permissible value for the hold attribute in the client request. If the client specifies a value higher than the gateway in the request, the gateway's value will be used. Otherwise, the value in the client request will be used.
<code>httpbind.inactivity</code>	180	The maximum time in seconds of client inactivity after which the gateway will terminate the connection to the client.
<code>httpbind.log4j.config</code>	Not applicable	The location of the log4j configuration file the gateway will use for logging. If you leave this parameter blank, then logging for the gateway is turned off. The logger name is httpbind (log4j.logger.httpbind) .
<code>httpbind.polling</code>	1 (second)	The minimum time, in seconds, a client must wait before sending another request.
<code>httpbind.requests</code>	2	The number of concurrent requests a client can make to the gateway. If the value of this parameter is less than the value for the JEP 124 hold attribute in the client request, the value for this parameter is set to hold+1 . Do not set this parameter to 1 , as doing so could severely degrade performance. See httpbind.hold for more information.
<code>httpbind.round_trip_delay</code>	1 (second)	The amount of time, in seconds, to allow in addition to time-outs for round trips to account for network latencies. Setting this value too high may degrade performance.

Table 34–1 (Cont.) XMPP/HTTP Gateway Configuration Parameters in httpbind.conf

Parameter	Default Value	Description
<code>httpbind.wait_time</code>	120 (seconds)	The default time, in seconds, within which the gateway will send a response to the client. If the client wait time is set to a value higher than the gateway wait time, the gateway's wait time is used.
<code>httpbind.component.password.cipher.delegate</code>	<code>com.sun.im.tools.passwordtool.Crypto</code>	If you want to use an encrypted password, this property and its default value are required.
<code>httpbind.component.password.cipher</code>	<code>com.sun.im.tools.passwordtool.CommsClientCipher</code>	If you want to use an encrypted password, this property and its default value are required.

Gateway Domain ID Key Parameters for httpbind.config

Table 34–2 describes the keys used to define each ID in the `httpbind.config` parameter. In each key described in the table, `gwdomain-id` is a domain identifier specified in `httpbind.config`.

Table 34–2 httpbind.config ID Keys

Key	Description
<code>gwdomain-id.domains</code>	Comma-separated list of domains for this ID.
<code>gwdomain-id.hosts</code>	Space-separated list of hosts for this ID. Each of these hosts must be able to service the domains listed in <code>gwdomain-id.domains</code> . This list helps provide failover across the domains. If no explicit route host mentioned in the request, one of the hosts listed in this key will be used to service that request.
<code>gwdomain-id.componentjid</code>	The component JID to use to connect to the host.
<code>gwdomain-id.password</code>	The password to use to connect to the host.
<code>gwdomain-id.multihosting</code>	If set to true , allows a packet destined to a domain, which is not preconfigured in <code>httpbind.conf</code> , to be sent to Instant Messaging Server. You use this parameter for a hosted domain setup. The default value for this parameter is true .