



Sun OpenSSO Enterprise 8.0 Installation and Configuration Guide



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

Part No: 820-3320
October 3, 2008

Copyright 2008 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. All rights reserved.

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

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

This distribution may include materials developed by third parties.

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

Sun, Sun Microsystems, the Sun logo, the Solaris logo, the Java Coffee Cup logo, docs.sun.com, Java, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. or its subsidiaries in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

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

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

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

Copyright 2008 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. Tous droits réservés.

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

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

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

Sun, Sun Microsystems, le logo Sun, le logo Solaris, le logo Java Coffee Cup, docs.sun.com, Java et Solaris sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc., ou ses filiales, aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

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

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

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

Contents

Preface	9
1 Getting Started With OpenSSO Enterprise 8.0	17
OpenSSO Enterprise 8.0 Requirements	18
Overview of Installing and Configuring OpenSSO Enterprise 8.0	19
Some OpenSSO Enterprise 8.0 Changes to Consider	19
Summary of the OpenSSO Enterprise 8.0 Installation and Configuration Steps	20
2 Installing OpenSSO Enterprise	23
Downloading OpenSSO Enterprise	23
Adding OpenSSO Enterprise Permissions to the Server Policy File	25
Deploying the OpenSSO Enterprise WAR File	28
▼ To Deploy the OpenSSO Enterprise WAR (opensso.war) File	28
Creating and Deploying Specialized OpenSSO Enterprise WAR Files	29
3 Configuring OpenSSO Enterprise Using the GUI Configurator	31
Starting the Configurator	31
▼ To Start the Configurator	31
Configuring OpenSSO Enterprise With the Default Configuration	33
▼ To Configure OpenSSO Enterprise With the Default Configuration	33
Configuring OpenSSO Enterprise With a Custom Configuration	34
▼ To Configure OpenSSO Enterprise With a Custom Configuration	34
4 Configuring OpenSSO Enterprise Using the Command-Line Configurator	43
Requirements to Run the Command-Line Configurator	43
Installing the Command-Line Configurator	43
▼ To Install the Command-Line Configurator	43

Configuring OpenSSO Enterprise Server	44
▼ To Configure OpenSSO Enterprise Using the Command-Line Configurator	44
OpenSSO Enterprise Configuration Parameters For the Command-Line Configurator	45
General and Server Parameters	45
Configuration Data Store Parameters	45
Multi-Server Deployment Parameters	46
User Data Store Parameters	47
Site Configuration Parameters	47
5 Installing the OpenSSO Enterprise Utilities and Scripts	49
Installing the OpenSSO Enterprise Utilities and Scripts in the ssoAdminTools.zip File	49
▼ To Install the OpenSSO Enterprise Utilities and Scripts in the ssoAdminTools.zip File ..	50
Running the Unix Authentication Helper (amunixd Daemon)	51
▼ To Run the Unix Authentication Helper (amunixd Daemon)	51
6 Implementing OpenSSO Enterprise Session Failover	53
Overview of OpenSSO Enterprise Session Failover	53
OpenSSO Enterprise Session Failover Components	53
OpenSSO Enterprise Session Failover Flow	56
Installing and Configuring the OpenSSO Enterprise Session Failover Components	56
Unzipping the ssoSessionTools.zip File	57
Running the Session Failover setup Script	58
Creating a New User to Connect to the Message Queue Broker (Optional)	59
Editing the amsessiondb Script (if Needed)	60
Encrypting the Message Queue Broker Password Using the amsfopassword Script (Required)	60
Running the amsfoscript to Start and Stop the Session Failover Components	61
Configuring Session Failover in the OpenSSO Enterprise Console	64
▼ To Configure Session Failover in the OpenSSO Enterprise Console	64
7 Deploying a Distributed Authentication UI Server	67
Distributed Authentication UI Server Overview	67
Distributed Authentication UI Server Deployment Scenario	67
Requirements for a Distributed Authentication UI Server Deployment	68
Flow for a Distributed Authentication End-User Request	69

Generating a Distributed Authentication UI Server WAR File	70
▼ To Generate a Distributed Authentication UI Server WAR File	70
Deploying the Distributed Authentication UI Server WAR File	71
▼ To Deploy the Distributed Authentication UI Server WAR File	71
Configuring the Distributed Authentication UI Server	71
▼ To Configure the Distributed Authentication UI Server	72
Accessing the Distributed Authentication User Interface Web Application	73
8 Deploying the Identity Provider (IDP) Discovery Service	75
Generating an IDP Discovery Service WAR File	75
▼ To Generate an IDP Discovery Service WAR File	75
Configuring the IDP Discovery Service	76
▼ To Configure the IDP Discovery Service	76
9 Installing the OpenSSO Enterprise Console Only	79
Requirements to Deploy Only the Console	79
Generating a Console Only WAR File	79
▼ To Generate a Console Only WAR File	79
Deploying and Configuring the Console Only WAR File	80
▼ To Deploy and Configure the Console Only WAR File	80
Accessing the Console	82
10 Installing OpenSSO Enterprise Server Only	83
Requirements to Deploy OpenSSO Enterprise Server Only	83
Generating a WAR File to Deploy OpenSSO Enterprise Server Only	83
▼ To Generate a WAR File to Deploy OpenSSO Enterprise Server Only	83
Deploying OpenSSO Enterprise Server Only	84
▼ To Deploy OpenSSO Enterprise Server Only	84
11 Installing the OpenSSO Enterprise Client SDK	87
OpenSSO Enterprise Client SDK Requirements	87
Installing the OpenSSO Enterprise Client SDK	88
▼ To Install the OpenSSO Enterprise Client SDK	88
Compiling and Running the Client SDK Samples	89

▼ To Compile and Run the Client SDK Samples	89
12 Configuring OpenSSO Enterprise Sessions	91
Setting Session Quota Constraints	91
Deployment Scenarios for Session Quota Constraints	91
Multiple Settings For Session Quotas	92
Configuring Session Quota Constraints	93
Configuring Session Property Change Notifications	94
▼ To Configure Session Property Change Notifications	95
13 Enabling the Access Manager (AMSDK) Identity Repository Plug-in	97
Requirements to Enable the Access Manager Identity Repository Plug-in	97
Configuring Sun Java System Directory Server	98
▼ To Configure an Existing Directory Server Identity Repository	98
▼ To Configure a New Directory Server Identity Repository	99
Configuring OpenSSO Enterprise Server	99
▼ To Configure OpenSSO Enterprise Server Using the <code>ssoadm</code> Command	100
Configure OpenSSO Enterprise Server Manually	100
Creating a Data Store Using the IdRepo Plug-in	107
▼ To Create a Data Store Using the IdRepo Plug-in	107
14 Managing LDAP Persistent Searches	109
Creation of Persistent Searches	109
Disabling Persistent Searches	110
▼ To Disable Persistent Searches Using the Console	111
Disabling Persistent Searches by Setting the <code>com.sun.am.event.connection.disable.list</code> Property	112
Re-Enabling Persistent Searches	112
Configuration Properties That Affect Persistent Searches	113
15 Customizing OpenSSO Enterprise Administration Console Pages	115
Customizing the OpenSSO Enterprise Login and Logout Pages	115
▼ To Customize the OpenSSO Enterprise Login and Logout Pages	115

16	Loading the OpenSSO Schema into Sun Java System Directory Server	117
	Loading the OpenSSO Schema into Directory Server	118
	▼ To Load the OpenSSO Schema into Directory Server	118
17	Uninstalling OpenSSO Enterprise	121
	Uninstalling OpenSSO Enterprise Server	121
	▼ To Uninstall OpenSSO Enterprise Server	121
	Uninstalling the OpenSSO Enterprise Utilities and Scripts	122
	▼ To Uninstall the OpenSSO Enterprise Utilities and Scripts	122
	Uninstalling a Distributed Authentication UI Server Deployment	122
	▼ To Uninstall a Distributed Authentication UI Server Deployment	122
	Uninstalling an IDP Discovery Deployment	123
	▼ To Uninstall an IDP Discovery Deployment	123
	Uninstalling a Client Sample Deployment	123
	▼ To Uninstall a Client Sample Deployment	123
	Uninstalling a Fedlet Deployment	124
	▼ To Uninstall a Fedlet Deployment	124
	Uninstalling an OpenSSO Enterprise Console Only Deployment	124
	▼ To Uninstall an OpenSSO Enterprise Console Only Deployment	124
	Uninstalling the OpenSSO Enterprise Client SDK	125
	▼ To Uninstall the OpenSSO Enterprise Client SDK	125
	Removing OpenSSO Enterprise Entries From Directory Server	125
	▼ To Remove OpenSSO Enterprise Entries From Directory Server	125
	Index	127

Preface

The *Sun™ OpenSSO Enterprise 8.0 Installation and Configuration Guide* describes how to install and configure OpenSSO Enterprise 8.0, including OpenSSO Enterprise server, server only (no administration console), administration console only, client SDK only, scripts and utilities, Distributed Authentication UI server, and a session failover deployment.

Contents

- “Who Should Use This Guide” on page 9
- “Before You Read This Guide” on page 9
- “How This Guide Is Organized” on page 10
- “Related Documentation” on page 10
- “Searching Sun Product Documentation” on page 12
- “Related Third-Party Web Site References” on page 12
- “Documentation, Support, and Training” on page 13
- “Typographic Conventions” on page 13
- “Default Paths and Directory Names” on page 14
- “Revision History” on page 15
- “Sun Welcomes Your Comments” on page 15

Who Should Use This Guide

This guide is intended for system administrators, system integrators, and others who are installing and configuring OpenSSO Enterprise.

Before You Read This Guide

Readers should be familiar with the following components and concepts:

- OpenSSO Enterprise technical concepts, as described in the *OpenSSO Enterprise 8.0 Technical Overview*
- Deployment platform: Solaris™, Linux, or Windows operating system
- Web container that will run OpenSSO Enterprise, such as Sun Java System Application Server, Sun Java System Web Server, BEA WebLogic, or IBM WebSphere Application Server

- Technical concepts: Lightweight Directory Access Protocol (LDAP), Java™ technology, JavaServer Pages™ (JSP™) technology, HyperText Transfer Protocol (HTTP), HyperText Markup Language (HTML), and eXtensible Markup Language (XML)

How This Guide Is Organized

This guide is organized as follows:

- Chapter 1, “Getting Started With OpenSSO Enterprise 8.0”
- Chapter 2, “Installing OpenSSO Enterprise”
- Chapter 3, “Configuring OpenSSO Enterprise Using the GUI Configurator”
- Chapter 5, “Installing the OpenSSO Enterprise Utilities and Scripts”
- Chapter 6, “Implementing OpenSSO Enterprise Session Failover”
- Chapter 7, “Deploying a Distributed Authentication UI Server”
- Chapter 9, “Installing the OpenSSO Enterprise Console Only”
- Chapter 10, “Installing OpenSSO Enterprise Server Only”
- Chapter 11, “Installing the OpenSSO Enterprise Client SDK”
- Chapter 12, “Configuring OpenSSO Enterprise Sessions”
- Chapter 13, “Enabling the Access Manager (AMSDK) Identity Repository Plug-in”
- Chapter 14, “Managing LDAP Persistent Searches”
- Chapter 17, “Uninstalling OpenSSO Enterprise”

Related Documentation

Related documentation is available as follows:

- “OpenSSO Enterprise Documentation Set” on page 10
- “Policy Agent Documentation” on page 11
- “Related Product Documentation” on page 12

OpenSSO Enterprise Documentation Set

The following table describes the OpenSSO Enterprise documentation set.

TABLE P-1 OpenSSO Enterprise Documentation Set

Title	Description
<i>Sun OpenSSO Enterprise 8.0 Release Notes</i>	Describes new features, installation notes, and known issues and limitations. The Release Notes are updated periodically after the initial release to describe any new features, patches, or problems.

TABLE P-1 OpenSSO Enterprise Documentation Set (Continued)

Title	Description
<i>Sun OpenSSO Enterprise 8.0 installation and Configuration Guide</i> (this guide)	Provides information about installing and configuring OpenSSO Enterprise, including OpenSSO Enterprise server, Administration Console only, client SDK, scripts and utilities, Distributed Authentication UI server, and session failover.
<i>Sun OpenSSO Enterprise 8.0 Technical Overview</i>	Provides an overview of how components work together to consolidate access control functions, and to protect enterprise assets and web-based applications. It also explains basic concepts and terminology.
<i>Sun OpenSSO Enterprise 8.0 Deployment Planning Guide</i>	Provides planning and deployment solutions for OpenSSO Enterprise.
<i>Sun OpenSSO Enterprise 8.0 Administration Guide</i>	Describes how to use the OpenSSO Enterprise Administration Console as well as how to manage user and service data using the command-line interface (CLI).
<i>Sun OpenSSO Enterprise 8.0 Administration Reference</i>	Provides reference information for the OpenSSO Enterprise command-line interface (CLI), configuration attributes, log files, and error codes.
<i>Sun OpenSSO Enterprise 8.0 Developer's Guide</i>	Provides information about customizing OpenSSO Enterprise and integrating its functionality into an organization's current technical infrastructure. It also provides details about the programmatic aspects of the product and its API.
<i>Sun OpenSSO Enterprise 8.0 C API Reference for Application and Web Policy Agent Developers</i>	Provides summaries of data types, structures, and functions that make up the public OpenSSO Enterprise C APIs.
<i>Sun OpenSSO Enterprise 8.0 Java API Reference</i>	Provides information about the implementation of Java packages in OpenSSO Enterprise.
<i>Sun OpenSSO Enterprise 8.0 Upgrade Guide</i>	Describes how to upgrade Sun Java System Access Manager and Sun Java System Federation Manager (including configuration data in Sun Java System Directory Server) to Sun OpenSSO Enterprise 8.0.
<i>Sun OpenSSO Enterprise 8.0 Performance Tuning Guide</i>	Provides information about how to tune OpenSSO Enterprise and its related components for optimal performance.

Policy Agent Documentation

Policy agent documentation includes these collections:

- 2.2 Policy Agents: <http://docs.sun.com/coll/1322.1>
- 3.0 Policy Agents: <http://docs.sun.com/coll/1809.1>

Related Product Documentation

The following table provides links to documentation collections for related products.

TABLE P-2 Related Product Documentation

Product	Link
Sun Java System Directory Server 6.3	http://docs.sun.com/coll/1224.4
Sun Java System Web Server 7.0 Update 3	http://docs.sun.com/coll/1653.3
Sun Java System Application Server 9.1	http://docs.sun.com/coll/1343.4
Sun Java System Message Queue 4.1	http://docs.sun.com/coll/1307.3
Sun Java System Web Proxy Server 4.0.6	http://docs.sun.com/coll/1311.6
Sun Java System Identity Manager 7.1	http://docs.sun.com/coll/1514.3

Searching Sun Product Documentation

Besides searching Sun product documentation from the docs.sun.comSM web site, you can use a search engine by typing the following syntax in the search field:

search-term site:docs.sun.com

For example, to search for “broker,” type the following:

broker site:docs.sun.com

To include other Sun web sites in your search (for example, java.sun.com, www.sun.com, and developers.sun.com), use sun . com in place of docs . sun . com in the search field.

Related Third-Party Web Site References

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

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

Documentation, Support, and Training

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

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

Typographic Conventions

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

TABLE P-3 Typographic Conventions

Typeface	Meaning	Example
AaBbCc123	The names of commands, files, and directories, and onscreen computer output	Edit your .login file. Use <code>ls -a</code> to list all files. <code>machine_name%</code> you have mail.
AaBbCc123	What you type, contrasted with onscreen computer output	<code>machine_name% su</code> Password:
<i>aabbcc123</i>	Placeholder: replace with a real name or value	The command to remove a file is <i>rm filename</i> .
<i>AaBbCc123</i>	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the <i>User's Guide</i> . <i>A cache</i> is a copy that is stored locally. Do <i>not</i> save the file. Note: Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX® system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE P-4 Shell Prompts

Shell	Prompt
C shell	machine_name%
C shell for superuser	machine_name#
Bourne shell and Korn shell	\$
Bourne shell and Korn shell for superuser	#

Default Paths and Directory Names

The OpenSSO Enterprise documentation uses the following terms to represent default paths and directory names:

TABLE P-5 Default Paths and Directory Names

Term	Description
<i>zip-root</i>	Represents the directory where the opensso.zip file is unzipped.
<i>OpenSSO-Deploy-base</i>	<p>Represents the deployment directory where the web container deploys the opensso.war file.</p> <p>This value varies depending on the web container. To determine the value of <i>OpenSSO-Deploy-base</i>, view the file name in the .openssocfg directory, which resides in the home directory of the user who deployed the opensso.war file. For example, consider this scenario with Application Server 9.1 as the web container:</p> <ul style="list-style-type: none">■ Application Server 9.1 is installed in the default directory: /opt/SUNWappserver.■ The opensso.war file is deployed by super user (root) on Application Server 9.1. <p>The .openssocfg directory is in the root home directory (/), and the file name in .openssocfg is:</p> <p>AMConfig_opt_SUNWappserver_domains_domain1_applications_j2ee-modules_opensso_</p> <p>Then, the value for <i>OpenSSO-Deploy-base</i> is:</p> <p>/opt/SUNWappserver/domains/domain1/applications/j2ee-modules/opensso</p>
<i>ConfigurationDirectory</i>	<p>Represents the name of the configuration directory specified during the initial configuration of OpenSSO Enterprise server instance using the Configurator.</p> <p>The default is opensso in the home directory of the user running the Configurator. Thus, if the Configurator is run by root, <i>ConfigurationDirectory</i> is /opensso.</p>

Revision History

TABLE P-6 Revision History

Date (Part Number)	Description of Change
October 3, 2008 (820-3320-05)	In-progress RR review draft
August 6, 2008	Early Access (EA) release draft

Sun Welcomes Your Comments

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

To share your comments, go to <http://docs.sun.com> and click Send comments. In the online form, provide the document title and part number. The part number is a seven-digit or nine-digit number that can be found on the title page of the guide or at the top of the document.

For example, the title of this guide is the *Sun OpenSSO Enterprise 8.0 Installation and Configuration Guide*, and the part number is 820-3320.

Getting Started With OpenSSO Enterprise 8.0

Sun™ OpenSSO Enterprise 8.0 is being developed as part of the OpenSSO project (<https://opensso.dev.java.net/>) and is the Sun commercial version of OpenSSO server.

OpenSSO Enterprise includes features such as access management, federation management, and web services security that are found in earlier releases of Sun Java System Access Manager and Sun Java System Federation Manager. However, OpenSSO Enterprise also includes many new features, which are described in the *OpenSSO Enterprise 8.0 Release Notes* and the *OpenSSO Enterprise 8.0 Technical Overview*.

OpenSSO Enterprise is available as a web archive (WAR) file on the OpenSSO project site: <http://opensso.dev.java.net/>

Before you install and configure OpenSSO Enterprise:

- First, check the “[OpenSSO Enterprise 8.0 Requirements](#)” on page 18.
- Then, review the “[Overview of Installing and Configuring OpenSSO Enterprise 8.0](#)” on page 19 before you continue with the detailed steps in subsequent chapters of this guide.

OpenSSO Enterprise 8.0 Requirements

TABLE 1–1 OpenSSO Enterprise 8.0 Requirements

Requirement	Description
Web container	<p>One of the following web containers must be running on the host server where you plan to deploy OpenSSO Enterprise:</p> <ul style="list-style-type: none">■ Sun Java System Application Server 9.1 Update 1 and Update 2■ Glassfish Application Server V2 UR1 and UR2■ Sun Java System Web Server 7.0 Update 3■ Apache Tomcat 5.5.x and 6.x■ BEA WebLogic Server 10■ BEA WebLogic Server 9.2 MP2■ Oracle Application Server 10g■ IBM WebSphere Application Server 6.1■ Apache Geronimo Application Server 2.1.2 (with Tomcat on Solaris systems only)■ JBoss Application Server 4.x <p>Note: These web container versions and any subsequent updates to the version are supported.</p> <p>For more information about supported versions and open issues for each web container, see the <i>OpenSSO Enterprise 8.0 Release Notes</i>.</p>
Configuration Data Store	<p>OpenSSO Enterprise requires a data store for its configuration data, which you select when you run the Configurator:</p> <ul style="list-style-type: none">■ Sun Java System Directory Server■ OpenSSO data store
User Data Store	<p>OpenSSO Enterprise also requires a data store for its user data:</p> <ul style="list-style-type: none">■ Sun Java System Directory Server <p>If you are deploying multiple OpenSSO Enterprise instances in a multiple server deployment, all instances must access the same Directory Server.</p> <ul style="list-style-type: none">■ Microsoft Active Directory■ IBM Tivoli Directory Server■ OpenSSO data store <p>Note: Storing user data in the OpenSSO data store is recommended only for prototype, proof of concept (POC), or developer deployments that have a small number of users. It is not recommended for production deployments.</p>

TABLE 1-1 OpenSSO Enterprise 8.0 Requirements (Continued)

Requirement	Description
Password encryption key	<p>If you deploying OpenSSO Enterprise in a multiple server deployment, you must use the same password encryption key value for each OpenSSO Enterprise instance.</p> <p>Copy the encryption key value from the first instance and then use this value when you configure each additional instance.</p>
Web container runtime user permissions	<p>If the runtime user of the OpenSSO Enterprise web container instance is a non-root user, this user must be able to write to its own home directory.</p> <p>For example, if you are installing Sun Java System Web Server, the default runtime user for the Web Server instance is <code>webserverd</code>. On Solaris systems, the <code>webserverd</code> user has the following entry in the <code>/etc/passwd</code> file:</p> <pre>webserverd:x:80:80:WebServer Reserved UID:/:</pre> <p>The <code>webserverd</code> user does not have permission to write to its default home directory (<code>/</code>). Therefore, you must change the permissions to allow the <code>webserverd</code> user to write to its default home directory. Otherwise, the <code>webserverd</code> user will encounter an error after you configure OpenSSO Enterprise using the Configurator.</p>
Mode	OpenSSO Enterprise is always deployed in Realm Mode.

Overview of Installing and Configuring OpenSSO Enterprise 8.0

- [“Some OpenSSO Enterprise 8.0 Changes to Consider” on page 19](#)
- [“Summary of the OpenSSO Enterprise 8.0 Installation and Configuration Steps” on page 20](#)

Some OpenSSO Enterprise 8.0 Changes to Consider

Before you install and configure OpenSSO Enterprise, here are a few changes to consider:

- You install OpenSSO Enterprise from a WAR file, using the web container administration console or deployment command. You no longer run a standalone installer.
- Configuration data, including policy agent configuration data, is stored in a centralized repository. This repository can be either Sun Java System Directory Server or the OpenSSO configuration data store, which is usually transparent to the user. OpenSSO Enterprise does not use the `AMConfig.properties` or `serverconfig.xml` files (except for compatibility with previous versions of Access Manager).

- You initially configure OpenSSO Enterprise using the Configurator. Then, to perform additional configuration, you use either the Administration Console or command-line utilities such as the new `ssoadm` utility. You no longer run the `amconfig` script with the `amsamplesilent` file.

Summary of the OpenSSO Enterprise 8.0 Installation and Configuration Steps

To install and configure an instance of OpenSSO Enterprise server, follow these general steps:

1. If necessary, install, configure, and start one of the supported web containers listed in [Table 1-1](#).
2. Download and unzip the `opensso.zip` file from the OpenSSO project site:
<http://opensso.dev.java.net/public/use/index.html>
Be sure to check the *OpenSSO Enterprise 8.0 Release Notes* page for any current issues.
3. Deploy the `opensso.war` file to the web container, using the web container administration console or deployment command.

For the detailed steps, see [Chapter 2, “Installing OpenSSO Enterprise.”](#)

4. Run either the GUI Configurator or command-line Configurator.
To run the GUI Configurator, enter the following URL in your browser:

protocol://host.domain:port/deploy_uri

For example: `http://opensso.example.com:8080/opensso`

If you are running the GUI Configurator, enter values in the Configurator fields or accept the default value for some fields. The Configurator has two configuration options:

- The **Default Configuration** option requires you to enter only the OpenSSO Enterprise administrator (`amAdmin`) and default policy agent (`UrlAccessAgent`) passwords. The Configurator then uses default values for the other configuration options.
Use the Default Configuration for development environments or simple demonstration purposes when you just want to evaluate OpenSSO Enterprise features.
- The **Custom Configuration** option allows you to enter specific configuration values for your deployment (or accept the default values).
Use the Custom Configuration for production and more complex environments. For example, a multi-server installation with several OpenSSO Enterprise instances behind a load balancer.

For the detailed steps, see [Chapter 3, “Configuring OpenSSO Enterprise Using the GUI Configurator,”](#) or [Chapter 4, “Configuring OpenSSO Enterprise Using the Command-Line Configurator.”](#)

5. Launch OpenSSO Enterprise using the specific web container console or deployment command, or by specifying the URL from Step 4 in your browser.
6. Login to the Console as the OpenSSO Enterprise administrator (amAdmin) using the password you specified when you ran the Configurator.
7. To make additional configuration changes to your deployment, use the OpenSSO Enterprise Administration Console or the `ssoadm` command-line utility. For information, refer to the Administration Console Online Help or the [Sun OpenSSO Enterprise 8.0 Administration Reference](#).

Installing OpenSSO Enterprise

Installing Sun™ OpenSSO Enterprise from a web archive (WAR) file involves these steps:

- “Downloading OpenSSO Enterprise” on page 23
- “Adding OpenSSO Enterprise Permissions to the Server Policy File” on page 25
- “Deploying the OpenSSO Enterprise WAR File” on page 28
- “Creating and Deploying Specialized OpenSSO Enterprise WAR Files” on page 29

Before you begin, check the “OpenSSO Enterprise 8.0 Requirements” on page 18.

Downloading OpenSSO Enterprise

OpenSSO Enterprise is available in the `opensso.zip` file, which you can download from the OpenSSO project site:

<http://opensso.dev.java.net/public/use/index.html>

The following table describes the layout after you unzip the `opensso.zip` file. The directory where you unzip the file is represented by *zip-root*.

TABLE 2-1 OpenSSO Enterprise opensso.zip File Layout

<i>zip-root/opensso/</i> Directory	Description
deployable-war	<p>OpenSSO Enterprise WAR and related files:</p> <ul style="list-style-type: none">■ opensso.war contains all OpenSSO Enterprise components. Use this file you to deploy OpenSSO Enterprise server or to generate specialized WAR files. See “Deploying the OpenSSO Enterprise WAR File” on page 28.■ console contains the additional files for deploying only the OpenSSO Enterprise Console. See Chapter 9, “Installing the OpenSSO Enterprise Console Only.”■ distauth contains the additional files for deploying a Distributed Authentication UI server. See Chapter 7, “Deploying a Distributed Authentication UI Server.”■ idpdiscovery contains the additional files for deploying OpenSSO Enterprise as an identity provider (IDP) using the Discovery Service. See Chapter 8, “Deploying the Identity Provider (IDP) Discovery Service.”■ noconsole contains the additional files for deploying OpenSSO Enterprise server without the Console. See Chapter 10, “Installing OpenSSO Enterprise Server Only.”■ fam-distauth.list, fam-console.list, fam-noconsole.list, fam-idpdiscovery.list, and fam-nosample.list lists the files that allow you to create specialized WAR files. See “Creating and Deploying Specialized OpenSSO Enterprise WAR Files” on page 29.
docs	Java API reference documentation (opensso-public-javadocs.jar).
integrations	<ul style="list-style-type: none">■ cleartrust contains the files to install and configure a custom authentication module that enables the SSO integration between OpenSSO Enterprise and RSA Access Manager (formerly RSA ClearTrust).■ oracle contains the files for integrating OpenSSO Enterprise with Oracle Access Manager (formerly Oblix).■ siteminder contains the files for integrating OpenSSO Enterprise with Computer Associates SiteMinder.
fedlet	Fedlet-unconfigured.zip file. After you unzip this file, fedlet.war allows you to deploy the Fedlet, a light-weight SAMLv2 service provider (SP). Follow the instructions in the Readme to configure the Fedlet metadata and COT and to deploy fedlet.war.

TABLE 2-1 OpenSSO Enterprise opensso.zip File Layout (Continued)

<i>zip-root/opensso/</i> Directory	Description
ldif	LDIF files for Sun Java System Directory Server, Microsoft Active Directory, and other LDAPv3 compliant directory servers.
libraries	DLL and JAR files for components such as OpenSSO Enterprise client SDK, the C SDK library for web policy agents, and the Secure Attribute Exchange (SAE) also known as Virtual Federation Proxy.
patches	Reserved for OpenSSO Enterprise patches.
samples	Client SDK and samples (opensso-client.zip). See Chapter 11, “Installing the OpenSSO Enterprise Client SDK.”
tools	OpenSSO Enterprise tools and utilities: <ul style="list-style-type: none"> ■ openssoAdminTools.zip contains files to setup and run the OpenSSO Enterprise command-line (CLI) utilities and scripts such as ssoadm and ampassword. ■ openssoSessionTools.zip contains the files to setup and configure OpenSSO Enterprise session failover. ■ helpers contains files for the UNIX authentication helper (amunixd). See Chapter 5, “Installing the OpenSSO Enterprise Utilities and Scripts.”
upgrade	Upgrade scripts and related files to upgrade Access Manager or Federation Manager. See the Sun OpenSSO Enterprise 8.0 Upgrade Guide .
xml	OpenSSO Enterprise XML files, such as amAdminConsole.xml, amAuth.xml, amSession.xml, and amUser.xml.

Adding OpenSSO Enterprise Permissions to the Server Policy File

If the Java security manager is enabled for the OpenSSO Enterprise web container, you must add OpenSSO Enterprise permissions to the web container server policy file. Usually, this file is named `server.policy`, but some web containers might use a different name.

If the Java security manager is disabled for the web container (for example, Glassfish), you do not need to add the permissions.



Caution – Before you modify the server policy file, backup the existing file.

The following examples show the permissions required for different web containers.

EXAMPLE 2-1 Application Server 9.1: OpenSSO Enterprise Permissions in the Server Policy File

```
// ADDITIONS FOR Access Manager
grant {
    permission java.net.SocketPermission "*", "listen,connect,accept,resolve";
    permission java.util.PropertyPermission "*", "read, write";
    permission java.lang.RuntimePermission "modifyThreadGroup";
    permission java.lang.RuntimePermission "setFactory";
    permission java.lang.RuntimePermission "accessClassInPackage.*";
    permission java.util.logging.LoggingPermission "control";
    permission java.lang.RuntimePermission "shutdownHooks";
    permission javax.security.auth.AuthPermission "getLoginConfiguration";
    permission javax.security.auth.AuthPermission "setLoginConfiguration";
    permission javax.security.auth.AuthPermission "modifyPrincipals";
    permission javax.security.auth.AuthPermission "createLoginContext.*";
    permission java.io.FilePermission "<<ALL FILES>>", "read,write,execute,delete";
    permission java.util.PropertyPermission "java.util.logging.config.class", "write";
    permission java.security.SecurityPermission "removeProvider.SUN";
    permission java.security.SecurityPermission "insertProvider.SUN";
    permission javax.security.auth.AuthPermission "doAs";
    permission java.util.PropertyPermission "java.security.krb5.realm", "write";
    permission java.util.PropertyPermission "java.security.krb5.kdc", "write";
    permission java.util.PropertyPermission "java.security.auth.login.config", "write";
    permission java.util.PropertyPermission "user.language", "write";
    permission javax.security.auth.kerberos.ServicePermission "*", "accept";
    permission javax.net.ssl.SSLPermission "setHostnameVerifier";
    permission java.security.SecurityPermission "putProviderProperty.IAIK";
    permission java.security.SecurityPermission "removeProvider.IAIK";
    permission java.security.SecurityPermission "insertProvider.IAIK";
    permission java.lang.RuntimePermission "setDefaultUncaughtExceptionHandler";
    permission javax.management.MBeanServerPermission "newMBeanServer";
    permission javax.management.MBeanPermission "*", "registerMBean";
    permission java.lang.RuntimePermission "createClassLoader";
    permission javax.security.auth.AuthPermission "getSubject";
    //following is already in AS 9.1EE but required for other containers
    permission javax.management.MBeanTrustPermission "register";
};
// END OF ADDITIONS FOR Access Manager
```

EXAMPLE 2-2 IBM WebSphere Application Server 6.1: OpenSSO Enterprise Permissions in the Server Policy File

```
// ADDITIONS FOR Access Manager
grant {
    permission java.net.SocketPermission "*", "listen,connect,accept,resolve";
    permission java.util.PropertyPermission "*", "read, write";
    permission java.lang.RuntimePermission "modifyThreadGroup";
    permission java.lang.RuntimePermission "setFactory";
```

EXAMPLE 2-2 IBM WebSphere Application Server 6.1: OpenSSO Enterprise Permissions in the Server Policy File *(Continued)*

```

    permission java.lang.RuntimePermission "accessClassInPackage.*";
    permission java.util.logging.LoggingPermission "control";
    permission java.lang.RuntimePermission "shutdownHooks";
    permission javax.security.auth.AuthPermission "getLoginConfiguration";
    permission javax.security.auth.AuthPermission "setLoginConfiguration";
    permission javax.security.auth.AuthPermission "modifyPrincipals";
    permission javax.security.auth.AuthPermission "createLoginContext.*";
    permission java.io.FilePermission "<<ALL FILES>>", "read,write,execute,delete";
    permission java.util.PropertyPermission "java.util.logging.config.class", "write";
    permission java.security.SecurityPermission "removeProvider.SUN";
    permission java.security.SecurityPermission "insertProvider.SUN";
    permission javax.security.auth.AuthPermission "doAs";
    permission java.util.PropertyPermission "java.security.krb5.realm", "write";
    permission java.util.PropertyPermission "java.security.krb5.kdc", "write";
    permission java.util.PropertyPermission "java.security.auth.login.config", "write";
    permission java.util.PropertyPermission "user.language", "write";
    permission javax.security.auth.kerberos.ServicePermission "*", "accept";
    permission javax.net.ssl.SSLPermission "setHostnameVerifier";
    permission java.security.SecurityPermission "putProviderProperty.IAIC";
    permission java.security.SecurityPermission "removeProvider.IAIC";
    permission java.security.SecurityPermission "insertProvider.IAIC";
    permission java.lang.RuntimePermission "setDefaultUncaughtExceptionHandler";
    permission javax.management.MBeanServerPermission "newMBeanServer";
    permission javax.management.MBeanPermission "*", "registerMBean";
    permission java.lang.RuntimePermission "createClassLoader";
    permission javax.security.auth.AuthPermission "getSubject";
    //following is already in AS 9.1EE but required for other containers
    permission javax.management.MBeanTrustPermission "register";
};
// END OF ADDITIONS FOR Access Manager

```

EXAMPLE 2-3 Glassfish Application Server: OpenSSO Enterprise Permissions in the Server Policy File

```

// ADDITIONS FOR Access Manager
grant {
    permission java.net.SocketPermission "*", "listen,connect,accept,resolve";
    permission java.util.PropertyPermission "*", "read, write";
    permission java.lang.RuntimePermission "modifyThreadGroup";
    permission java.lang.RuntimePermission "setFactory";
    permission java.lang.RuntimePermission "accessClassInPackage.*";
    permission java.util.logging.LoggingPermission "control";
    permission java.lang.RuntimePermission "shutdownHooks";
    permission javax.security.auth.AuthPermission "getLoginConfiguration";
    permission javax.security.auth.AuthPermission "setLoginConfiguration";
    permission javax.security.auth.AuthPermission "modifyPrincipals";

```

EXAMPLE 2-3 Glassfish Application Server: OpenSSO Enterprise Permissions in the Server Policy File
(Continued)

```

permission javax.security.auth.AuthPermission "createLoginContext.*";
permission java.io.FilePermission "<<ALL FILES>>", "read,write,execute,delete";
permission java.util.PropertyPermission "java.util.logging.config.class", "write";
permission java.security.SecurityPermission "removeProvider.SUN";
permission java.security.SecurityPermission "insertProvider.SUN";
permission javax.security.auth.AuthPermission "doAs";
permission java.util.PropertyPermission "java.security.krb5.realm", "write";
permission java.util.PropertyPermission "java.security.krb5.kdc", "write";
permission java.util.PropertyPermission "java.security.auth.login.config", "write";
permission java.util.PropertyPermission "user.language", "write";
permission javax.security.auth.kerberos.ServicePermission "*", "accept";
permission javax.net.ssl.SSLPermission "setHostnameVerifier";
permission java.security.SecurityPermission "putProviderProperty.IAIC";
permission java.security.SecurityPermission "removeProvider.IAIC";
permission java.security.SecurityPermission "insertProvider.IAIC";
permission java.lang.RuntimePermission "setDefaultUncaughtExceptionHandler";
permission javax.management.MBeanServerPermission "newMBeanServer";
permission javax.management.MBeanPermission "*", "registerMBean";
permission java.lang.RuntimePermission "createClassLoader";
permission javax.security.auth.AuthPermission "getSubject";
//following is already in AS 9.1EE but required for other containers
permission javax.management.MBeanTrustPermission "register";
};
// END OF ADDITIONS FOR Access Manager

```

Deploying the OpenSSO Enterprise WAR File

Deploy the OpenSSO Enterprise WAR (opensso.war) file using the web container administration console or deploy command.

▼ To Deploy the OpenSSO Enterprise WAR (opensso.war) File

1 Login as a user who has the following privileges:

- Access to the OpenSSO Enterprise web container administration console, if you plan to deploy opensso.war using the console.
- or
- The capability to execute the web container's deploy command-line utility, if you plan to deploy opensso.war using the CLI.

- 2 If necessary, copy `opensso.war` to the server where you want to deploy OpenSSO Enterprise.
- 3 Deploy `opensso.war` using either the web container administration console or deploy command.

If the OpenSSO Enterprise web container administration console includes the option to deploy a WAR file, this method is usually the simplest one to use.

Otherwise, use the web container deploy command. For example, the following command deploys `opensso.war` on the Application Server 9.1 web container on Solaris systems:

```
# cd /opt/SUNWappserver/appserver/bin
# ./asadmin deploy --user admin --passwordfile /tmp/pwdfilename
--port 4848 zip-root/opensso/deployable-war/opensso.war
```

where:

- `zip-root` is where you unzipped the `opensso.zip` file. Or, if you copied `opensso.war` to a different location, use that location in the command.
- `/tmp/pwdfilename` is the Application Server 9.1 password file. This ASCII text file contains the `AS_ADMIN_PASSWORD` variable set to the administrator password.

Next Steps Continue with the initial OpenSSO Enterprise server configuration using the Configurator:

[Chapter 3, “Configuring OpenSSO Enterprise Using the GUI Configurator”](#)

or

[Chapter 4, “Configuring OpenSSO Enterprise Using the Command-Line Configurator”](#)

Creating and Deploying Specialized OpenSSO Enterprise WAR Files

In addition to an OpenSSO Enterprise full server deployment, you can also create and deploy the following specialized WAR files:

- Distributed Authentication UI Server: [Chapter 7, “Deploying a Distributed Authentication UI Server”](#)
- IDP Discovery Service: [Chapter 8, “Deploying the Identity Provider \(IDP\) Discovery Service”](#)
- OpenSSO Enterprise Administration Console only: [Chapter 9, “Installing the OpenSSO Enterprise Console Only”](#)
- OpenSSO Enterprise server without the Administration Console: [Chapter 10, “Installing OpenSSO Enterprise Server Only”](#)
- OpenSSO Enterprise client SDK: [Chapter 11, “Installing the OpenSSO Enterprise Client SDK”](#)

Configuring OpenSSO Enterprise Using the GUI Configurator

Sun™ OpenSSO Enterprise includes the Configurator to perform the initial configuration of an OpenSSO Enterprise server instance. This chapter describes how to run the GUI Configurator, including:

- “Starting the Configurator” on page 31
- “Configuring OpenSSO Enterprise With the Default Configuration” on page 33
- “Configuring OpenSSO Enterprise With a Custom Configuration” on page 34

To run the Configurator from the command-line, see TBD.

Starting the Configurator

▼ To Start the Configurator

Before You Begin **Important:** If you plan to use Sun Java System Directory Server to store configuration or user data, Directory Server must be installed and running before you launch the Configurator.

1 Launch OpenSSO Enterprise.

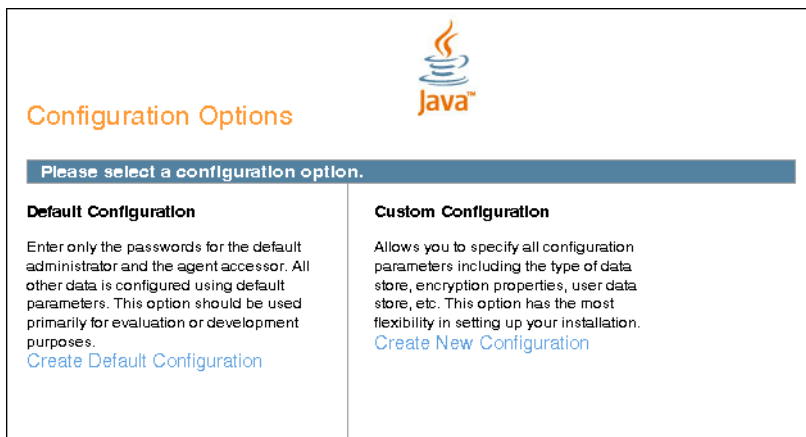
When you access OpenSSO Enterprise for the first time, you will be directed to the Configurator, to perform the OpenSSO Enterprise initial configuration.

To start the Configurator, specify the following URL in your browser:

protocol://host.domain:port/deploy_uri

For example: `http://opensso.example.com:8080/opensso`

The Configurator starts and display the **Configuration Options** page:



2 Select the configuration option:

- **Default Configuration:** You specify and confirm passwords for the OpenSSO Enterprise administrator (amAdmin) and the default policy agent user (UrlAccessAgent), which is the policy agent user that connects to OpenSSO Enterprise server. The Configurator uses default values for the other configuration settings.

Choose **Default Configuration** for development environments or simple demonstration purposes when you just want to evaluate OpenSSO Enterprise features.

Click **Create Default Configuration** and continue with “[Configuring OpenSSO Enterprise With the Default Configuration](#)” on page 33.

or

- **Custom Configuration:** You specify the configuration settings that meet the specific requirements for your deployment (or accept the default settings).

Choose **Custom Configuration** for production and more complex environments. For example, a multi-server installation with several OpenSSO Enterprise instances behind a load balancer.

Click **Create New Configuration** and continue with “[Configuring OpenSSO Enterprise With a Custom Configuration](#)” on page 34.

Configuring OpenSSO Enterprise With the Default Configuration

In this scenario, you launched the Configurator and clicked **Create Default Configuration**.

▼ To Configure OpenSSO Enterprise With the Default Configuration

- 1 On the **Default Configuration Options** page, enter and confirm the following passwords:
 - **Default User** (amAdmin) is the OpenSSO Enterprise administrator.
 - **Default Policy Agent** (UrlAccessAgent) is the policy agent user that connects to OpenSSO Enterprise server.

OpenSSO Configurator

Default Configuration Option

Use this option for a quick setup. Only the super user name and agent user name are required. All other configuration parameters are defaulted for you. The user and agent passwords must be different values. * Indicates required field

Default User [amAdmin]

* Password

* Confirm Password

Default Policy Agent [UrlAccessAgent]

* Password

* Confirm Password

Create Configuration Cancel

- 2 Click **Create Configuration** to continue.

Next Steps When the configuration is complete, the Configurator displays a link to the OpenSSO Enterprise Administration Console to perform any additional configuration required for your deployment.

If a problem occurred during the configuration, the Configurator displays an error message. If you can, correct the error and retry the configuration. Also, check the web container log files to help determine the problem.

Configuring OpenSSO Enterprise With a Custom Configuration

In this scenario, you launched the Configurator and clicked **Create New Configuration**.

▼ To Configure OpenSSO Enterprise With a Custom Configuration

- 1 On the **Default User Password** page, enter and confirm the `amAdmin` password:

OpenSSO Configurator

Custom Configuration Option

➔ **General**

2. Server Settings
3. Configuration Store
4. User Store
5. Site Configuration
6. Agent Information
7. Summary

Step 1: General

Enter the password for the default user, amAdmin. The password must be at least 8 characters in length. If this configuration will be part of an existing deployment, the password you enter must match that of the original deployment.

* Indicates required field

Default User Password

Default User [amAdmin]

* Password

* Confirm Password

Previous Next Cancel

Click **Next** to continue.

- 2 On the **Server Settings** page, specify the OpenSSO Enterprise server information:

OpenSSO Configurator

Custom Configuration Option

1. General
→ **Server Settings**
3. Configuration Store
4. User Store
5. Site Configuration
6. Agent Information
7. Summary

Step 2: Server Settings
Confirm the following settings to use for the server.

* Indicates required field

Server Settings

* Server URL: ☒ OK

* Cookie Domain: ☒ OK

* Platform Locale:

* Configuration Directory:

Previous Next Cancel

- **Server URL** is the host server where you deployed OpenSSO Enterprise. It can be one of the following values:
 - localhost
 - Fully qualified domain name (FQDN). For example: `http://host.example.com:8080`
If you plan to use the OpenSSO Enterprise client SDK or a policy agent, you must specify the FQDN.

The default is the host where you deployed the `opensso.war` file.

- **Cookie Domain** is the name of the trusted DNS domain that OpenSSO Enterprise returns to a browser when it grants a single sign-on (SSO) token to a user.
Specify a value only if the FQDN is used as the Server URL. For example, if the FQDN for Server URL is `http://host.example.com:8080`, the value is `.example.com`.
- **Platform Locale** is the default language subtype for OpenSSO Enterprise. The default is `en_US` (US English).
Other values can be `de` (German), `es` (Spanish), `fr` (French), `ja` (Japanese), `zh` (Chinese), or `zh_TW` (Simplified Chinese).
- **Configuration Directory** is the location of the OpenSSO Enterprise configuration directory.
Important: The runtime user of the OpenSSO Enterprise web container instance must have write access to the location where this directory will be created. For example, if the web container instance is running as the `websrvd` user, then the `websrvd` user must be able to write to the configuration directory.

Click **Next** to continue.

3 Specify the Configuration Data Store Settings:

Check whether the instance you are configuring is the **First Instance** (or the only instance) or if you want to **Add to an Existing Deployment**.

If you check **Add to Existing Deployment**, enter the **Server URL**.

The screenshot shows the 'OpenSSO Configurator' window with the 'Custom Configuration Option' tab selected. The left sidebar lists steps: 1. General, 2. Server Settings, 3. Configuration Store (selected), 4. User Store, 5. Site Configuration, 6. Agent Information, and 7. Summary. The main area is titled 'Step 3: Configuration Data Store Settings' and includes instructions: 'If no other OpenSSO instance already exists in the environment, then choose First Instance. If one or more OpenSSO instances already exist in the environment, choose Add to Existing Deployment.' Below this are two radio buttons: 'First Instance' (selected) and 'Add to Existing Deployment'. A note '* Indicates required field' is present. A 'Configuration Store Details' panel is shown with two radio buttons: 'OpenSSO' (selected) and 'Sun Java System Directory Server'. Below these are five fields: 'SSL Enabled' (checkbox), 'Host Name' (text box with 'localhost'), 'Port' (text box with '50389'), 'Encryption Key' (text box with '15rKFcBP3KihVnFqYV907XHWS9Xt'), and 'Root Suffix' (text box with 'dc=opensso,dc=java,dc=net'). At the bottom are 'Previous', 'Next', and 'Cancel' buttons.

Configuration Store Details:

- **Configuration Data Store:**
 - **OpenSSO** stores OpenSSO Enterprise configuration data under the *configuration_directory/openss* directory on the local server.
 - **Sun Java System Directory Server** stores OpenSSO Enterprise configuration data in Sun Java System Directory Server.
- **SSL Enabled:** Check if you want to use LDAPS to connect to the directory server hosting the configuration data store.
- **Host Name** is the directory server host name.
- **Port** is the directory server port number. Default is 50389.
- **Encryption Key** is a random number used to encrypt passwords. Either accept the default encryption key value or specify a new value. The encryption key must be at least 12 characters.

Important: If you are deploying multiple OpenSSO Enterprise instances in a multiple server deployment, you must use the same password encryption key value for each instance.

- **Root Suffix** is the directory server initial or root suffix.

Note – If you are configuring a second instance in an OpenSSO Enterprise site and the first instance in the site is SSL-enabled, you must import the root CA certificate of the server certificate of the first OpenSSO Enterprise instance into the second OpenSSO Enterprise instance's web container's JVM key store.

By default, the JDK key store is the `/jre/lib/security/cacerts` file. For example, to import a root CA certificate to this key store:

```
keytool -import -v -alias -keystore /jre/lib/security/cacerts -storepass  
changeit -file CAcert.crt
```

Then, to verify that the root CA certificate was stored correctly in the key store:

```
keytool -list -keystore /jre/lib/security/cacerts -storepass changeit
```

After you import the certificate, restart the web container for the second instance.

Click **Next** to continue.

4 Specify the User Data Store Settings:

- **OpenSSO User Data Store** stores user data in the OpenSSO user data store.

Note: Storing user data in the OpenSSO data store is recommended only for prototype, proof of concept (POC), or developer deployments that have a small number of users. It is not recommended for production deployments.

- **Other User Data Store** stores user data in a data store such as Sun Java System Directory Server, Microsoft Active Directory, or IBM Tivoli Directory Server.

OpenSSO Configurator

Custom Configuration Option

1. General
2. Server Settings
3. Configuration Store
→ 4. User Store
5. Site Configuration
6. Agent Information
7. Summary

Step 4: User Data Store Settings

You can use the data store that comes with the OpenSSO configuration data store, or you can use a different user data store. A good practice for setting up production environments is to use an external user data store, one that is different than the OpenSSO user data store. Please note that Policy Service and LDAP Authentication Module shall be configured to use the Directory Administrator DN and Password provided here.

☐ OpenSSO User Data Store
☒ Other User Data Store

* Indicates required field

User Store Details

*SSL Enabled ☐
 *Directory Name
 *Port
 *Root Suffix
 *Login ID
 *Password
 *User Data Store Type ☒ LDAP with OpenSSO Schema
☐ Generic LDAP (no OpenSSO Schema)

Previous Next Cancel

User Store Details:

- **SSL Enabled:** Check if you want to use LDAPS to connect to the directory server hosting the user data store.

Note – Before you continue with the configuration, the JVM of the web container instance on which OpenSSO Enterprise is deployed must trust the root CA certificate of the certificate on the LDAPS-enabled directory server. The root CA certificate for the directory server certificate must be imported into the web container JVM's trust store.

The default trust store is `/jre/lib/security/cacerts`. If this certificate is not imported, use the `keytool` utility to import the directory server root CA. For example, to import a root CA certificate to this key store:

```
keytool -import -v -alias -keystore /jre/lib/security/cacerts -storepass
changeit -file CACert.crt
```

Then, verify that the root CA certificate was stored correctly in the key store:

```
keytool -list -keystore /jre/lib/security/cacerts -storepass changeit
```

After you import the certificate, restart the web container.

- **Directory Name** is the hostname of the directory server that will serve as the user store.
- **Port** is the user directory server port number. Default is 389. If **SSL Enabled** is checked the **Port** value should be the LDAPS port of the directory server instance.

- **Root Suffix** is the user directory server initial or root suffix.
- **Login ID** is the user who has unlimited access the user directory server.
- **Password** is the password for the user specified in Login ID.
- **User Data Store Type:**
 - **LDAP with OpenSSO Schema:** The directory server already has the OpenSSO Enterprise schema loaded.
 - **Generic LDAP:** The directory server does not have the OpenSSO Enterprise schema loaded.

Click **Next** to continue.

- 5 On the **Site Configuration** page, specify whether this OpenSSO Enterprise instance will be deployed behind a load balancer as part of a site configuration.

If **No**, click **Next** to continue.

If **Yes**, specify the **Site Configuration Details**:

- **Site Name** is the name of the site.
- **Load Balancer URL** is the URL of the load balancer in the site.

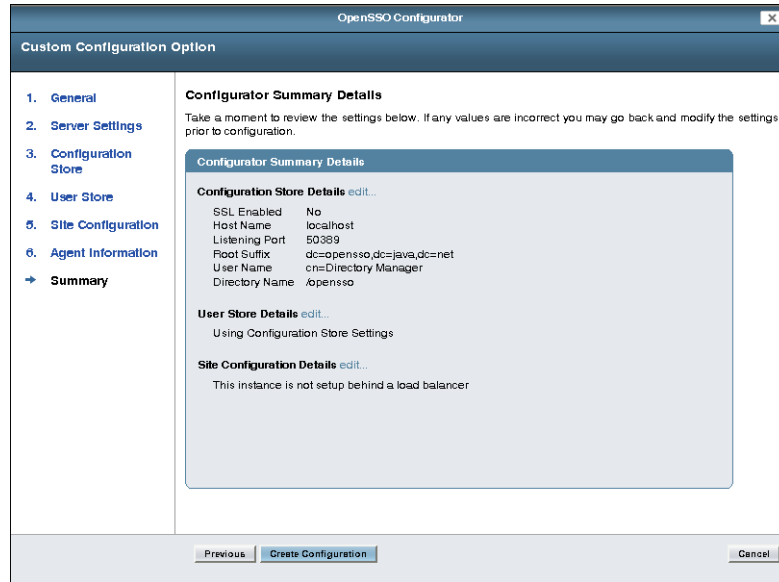
Click **Next** to continue.

- 6 Specify and confirm the password for the Default Policy Agent user (UrlAccessAgent):

The screenshot shows the 'OpenSSO Configurator' window with the 'Custom Configuration Option' tab selected. On the left, a navigation pane lists steps 1 through 7: General, Server Settings, Configuration Store, User Store, Site Configuration, Agent Information (highlighted with a blue arrow), and Summary. The main area displays 'Step 6: Default Policy Agent User' with a sub-header 'Default Policy Agent (UriAccessAgent)'. Below this, there are two text input fields labeled 'Password' and 'Confirm Password', both marked with a red asterisk to indicate they are required. A note above the fields states 'These settings are used by OpenSSO policy agents for retrieving policy agent properties.' and a legend indicates '* Indicates required field'. At the bottom of the window, there are three buttons: 'Previous', 'Next' (highlighted in blue), and 'Cancel'.

Click **Next** to continue.

7 Check the Summary page:



If the settings in the summary are correct, click **Create Configuration**.

To make changes, click **Previous** or **Edit** to return to previous pages to make changes to your configuration (or click **Cancel** to start over).

If a problem occurred during the configuration, the Configurator displays an error message. If you can, correct the error and retry the configuration.

Also, check the web container log files to help determine the problem. In some cases, there might be an `amSetupServlet` debug log (`/opensso/deploy_uri/debug/amSetupServlet`) containing errors or exceptions.

Next Steps When the configuration is complete, the Configurator displays a link to the OpenSSO Enterprise Administration Console so you can perform any additional configuration required for your deployment.

Login to the Console as `amAdmin` using the password you specified during the initial configuration using the Configurator.

The Console includes Common Tasks to help you configure common deployment scenarios. For information about the Common Tasks as well as other configuration tasks you can do in the Console, see the Console online Help.

Configuring OpenSSO Enterprise Using the Command-Line Configurator

To configure OpenSSO Enterprise server using the command-line Configurator, you set parameters in a configuration file and then run the Configurator from the command line using the configuration file as input. You can run the Configurator on the same system as OpenSSO Enterprise server or from a remote system.

Requirements to Run the Command-Line Configurator

The requirements to install and run the command-line Configurator include:

- You must have downloaded and unzipped the `opensso.zip` file, as described in [Chapter 2, “Installing OpenSSO Enterprise.”](#)
- The Configurator requires JDK 1.5 or later, so make sure that your `JAVA_HOME` environment variable points to JDK 1.5 or later.

Installing the Command-Line Configurator

After you unzip the `opensso.zip` file, the command-line Configurator and related files are in the following file:

`zip-root/opensso/tools/ssoConfiguratorTools.zip`

where *zip-root* is the directory where you unzipped `opensso.zip`.

▼ To Install the Command-Line Configurator

- 1 Log in as super user (`root`).
- 2 Change to the `zip-root/opensso/tools` directory.

3 Unzip the `ssoConfiguratorTools.zip` file to get these files:

- `README.setup` describes how to run the Configurator.
- `configurator.jar` contains the binary files (`OpenSSOConfigurator.class` and `OpenSSOConfigurator.properties`).
- `sampleconfiguration` is a sample input file that you edit before you run the Configurator.
- `license.txt` describes the Common Development and Distribution License (CDDL).

Remote system. If you plan to run the Configurator on a remote system, copy the `ssoConfiguratorTools.zip` file to the remote system before you unzip it.

Configuring OpenSSO Enterprise Server

▼ To Configure OpenSSO Enterprise Using the Command-Line Configurator

- 1 Make sure your `JAVA_HOME` environment variable points to JDK 1.5 or later.
- 2 Log in as super user (`root`).
- 3 Change to the directory where you unzipped the `ssoConfiguratorTools.zip` file.
- 4 Create a configuration file and set the properties required for your deployment.

Sun provides the OpenSSO Enterprise server configuration parameters in the `sampleconfiguration` file. Either edit `sampleconfiguration` and use it when you run the Configurator, or copy this file and edit the new file.

See [“OpenSSO Enterprise Configuration Parameters For the Command-Line Configurator” on page 45](#) for the properties you can set.

- 5 Run the Configurator. For example:

```
# java -jar configurator.jar -f configuration-file
```

where *configuration-file* contains the configuration properties you set in the previous step.

OpenSSO Enterprise Configuration Parameters For the Command-Line Configurator

- “General and Server Parameters” on page 45
- “Configuration Data Store Parameters” on page 45
- “Multi-Server Deployment Parameters” on page 46
- “User Data Store Parameters” on page 47
- “Site Configuration Parameters” on page 47

General and Server Parameters

- `SERVER_URL` is the URL of the web container on which OpenSSO Enterprise server is deployed. For example: `SERVER_URL=http://ssohost.example.com:58080`
- `DEPLOYMENT_URI` is the OpenSSO Enterprise server deployment URI. Default: `DEPLOYMENT_URI=/opensso`
- `BASE_DIR` is the configuration directory. Default: `BASE_DIR=/opensso`
- `PLATFORM_LOCALE` is the OpenSSO Enterprise server locale. Default: `locale=en_US`
The default is `en_US` (US English). Other values can be `de` (German), `es` (Spanish), `fr` (French), `ja` (Japanese), `zh` (Chinese), or `zh_TW` (Simplified Chinese).
- `AM_ENC_KEY` is the password encryption key. In a multi-server installation, this parameter must have the same value as the other servers. By default, `AM_ENC_KEY` is set to blank (`""`), which means that OpenSSO Enterprise server will generate a random password encryption key.

If you specify a password encryption key, the key must be at least 8 characters. If this configuration will be part of an existing deployment, the password encryption key you enter must match that of the original deployment.

- `ADMIN_PWD` is the password for the default OpenSSO Enterprise administrator, `amAdmin`. The password must be at least 8 characters in length. If this configuration will be part of an existing deployment, the password you enter must match that of the original deployment.
- `COOKIE_DOMAIN` is the name of the trusted DNS domain that OpenSSO Enterprise server returns to a browser when it grants a session ID to a user. For example: `COOKIE_DOMAIN=.example.com`
- `AMLDAPUSERPASSWD` is the password for default policy agent user [`UrlAccessAgent`].

Configuration Data Store Parameters

- `DATA_STORE` is the type of configuration data store. Values can be:
embedded - OpenSSO configuration data store

dirServer - Sun Java System Directory Server

For example: DATA_STORE=embedded

If dirServer is specified and the configuration data store contains the configuration of existing OpenSSO Enterprise servers, this OpenSSO Enterprise server will be added to the existing multi-server setup.

- DIRECTORY_SSL specifies if the configuration data store is using SSL. Values can be:
 - SSL: SSL is used.
 - SIMPLE: SSL is not used.

For example: DIRECTORY_SSL=SIMPLE

- DIRECTORY_SERVER is the fully qualified host name of the configuration data store. For example: DIRECTORY_SERVER=ds.example.com
- DIRECTORY_PORT is the port on which the configuration data store is listening for connections. For example: DIRECTORY_PORT=50389
- ROOT_SUFFIX is the initial or root suffix of the configuration data store. For example: ROOT_SUFFIX=dc=opensso,dc=java,dc=net
- DS_DIRMGRDN is the DN (distinguished name) of the directory manager, who is the user who has unrestricted access to the configuration data store. Default: DS_DIRMGRDN=cn=Directory Manager
- DS_DIRMGRPWD is the password for the directory manager of the configuration data store.

Multi-Server Deployment Parameters

- DS_EMB_REPL_FLAG is a flag that enables the configuration data store in a multi-server setup. This flag is valid only if DATA_STORE=embedded. To enable this flag, set the value to embReplFlag. For example: DS_EMB_REPL_FLAG=embReplFlag
- DS_EMB_REPL_REPLPORT1 is the replication port of the configuration data store of the new OpenSSO Enterprise server. For example: DS_EMB_REPL_REPLPORT1=58989
- DS_EMB_REPL_HOST2 is the host name of the existing OpenSSO Enterprise server. For example: DS_EMB_REPL_HOST2=host2.example.com
- DS_EMB_REPL_PORT2 is the listening port of the configuration data store of the existing OpenSSO Enterprise server. For example: DS_EMB_REPL_PORT2=50389
- DS_EMB_REPL_REPLPORT2 is the replication port of the configuration data store of the existing OpenSSO Enterprise server. For example: DS_EMB_REPL_REPLPORT2=50889

User Data Store Parameters

- `USERSTORE_TYPE` is the type of user data store. Values can be:
 - `LDAPv3ForAMD`: LDAP with OpenSSO Schema
 - `LDAPv3`: Generic LDAP (no OpenSSO Schema)
 - `''` (blank): The configuration data store will be the same as the user data store. `DATA_STORE` must be embedded. The remaining user data store properties will be ignored.
- `USERSTORE_SSL` specifies if the user data store is using SSL. Values can be:
 - `SSL`: SSL is used.
 - `SIMPLE`: SSL is not used.
- `USERSTORE_HOST` is the host name of the user data store. For example:
`ssohost.example.com`
- `USERSTORE_PORT` is the port on which the user data store is listening for connections. Default is 389.
- `USERSTORE_SUFFIX` is the initial or root suffix of the user data store. For example:
`dc=opensso,dc=java,dc=net`
- `USERSTORE_MGRDN` is the DN (distinguished name) of the directory manager, which is the user who has unrestricted access to the user data store. Default is `cn=Directory Manager`
- `USERSTORE_PASSWD` is the password for the directory manager of the user data store.

Site Configuration Parameters

- `LB_SITE_NAME` is the name of the site.
- `LB_PRIMARY_URL` is the load balancer URL. For example:
`http://lb.example.com:58080/opensso`

Installing the OpenSSO Enterprise Utilities and Scripts

The Sun™ OpenSSO Enterprise ZIP (`opensso.zip`) file includes utilities, scripts, libraries, and other supporting files in the following ZIP files:

- `ssoAdminTools.zip` contains the files to run the OpenSSO Enterprise command-line utilities and scripts such as `ssoadm`, `amtune`, and `ampassword`.
See [“Installing the OpenSSO Enterprise Utilities and Scripts in the `ssoAdminTools.zip` File” on page 49](#).
- `ssoSessionTools.zip` contains the scripts and supporting files to install Sun Java System Message Queue and the Oracle Berkeley DB, which then allows you to configure multiple OpenSSO Enterprise instances for session failover.
For information about the `ssoSessionTools.zip` file and how to configure session failover, see [Chapter 6, “Implementing OpenSSO Enterprise Session Failover.”](#)

This chapter also describes [“Running the Unix Authentication Helper \(`amunxd` Daemon\)” on page 51](#).

For information about uninstallation, see [“Uninstalling the OpenSSO Enterprise Utilities and Scripts” on page 122](#)

Installing the OpenSSO Enterprise Utilities and Scripts in the `ssoAdminTools.zip` File

After you download and unzip the `opensso.zip` file, the `ssoAdminTools.zip` file is available in the `zip-root/opensso/tools` directory.

The following table describes the layout after you unzip the `ssoAdminTools.zip` file. The directory where you unzip `ssoAdminTools.zip` is represented by *tools-zip-root*.

TABLE 5-1 openssoAdminTools.zip File Layout

<i>tools-zip-root</i> File or Directory	Description
README.setup	Description of the ssoAdminTools.zip file.
license.txt	License agreement.
setup	Script to install the tools on Solaris and Linux systems.
setup.bat	Script to install the tools on Windows systems.
lib/	JAR files required to run the scripts.
locale/	Properties files required to run the scripts.
mo/	Files for localizing the amtune scripts
template/	Script templates for Solaris, Linux, and Windows systems.

▼ To Install the OpenSSO Enterprise Utilities and Scripts in the ssoAdminTools.zip File

- 1 **Make sure that your `JAVA_HOME` environment variable points to JDK 1.5 or later.**
- 2 **Create a new directory to unzip the ssoAdminTools.zip file (represented by *tools-zip-root* in the previous table).**
- 3 **Unzip the ssoAdminTools.zip file in the new directory.**
- 4 **In the directory where you unzipped the ssoAdminTools.zip file, run the setup script:**
On Solaris and Linux systems, run the setup script as follows:

```
# ./setup
```

When you are prompted, enter the path to the OpenSSO Enterprise configuration directory. The configuration directory was specified during the initial configuration using the Configurator. For example: /opensso

Considerations:

On Windows systems, run the setup.bat script.

Next Steps You can now run the OpenSSO Enterprise CLI utilities and scripts from the following directory:

tools-zip-root/deploy_uri/bin

where:

- *tools-zip-root* is the directory where you unzipped the *ssoAdminTools.zip* file.
- *deploy_uri* is the name of the OpenSSO Enterprise deploy URI. For example: *opensso*

For information about the CLI utilities, see the *OpenSSO Enterprise 8.0 Administration Reference*.

For information about the tuning scripts, see the *OpenSSO Enterprise 8.0 Performance and Tuning Guide*.

Running the Unix Authentication Helper (amunxd Daemon)

The Unix authentication module is supported on Solaris SPARC, Solaris x86, or Linux systems. The Unix authentication module requires the *amunxd* helper daemon for Unix authentication.

After you unzip the *opensso.zip* file, the helper files for the Unix authentication module are in the *zip-root/opensso/tools/helpers* directory.

▼ To Run the Unix Authentication Helper (amunxd Daemon)

- 1 To change any of the Unix authentication module configuration values, use the OpenSSO Enterprise administration Console:
 - a. Login into the Console as *amadmin*.
 - b. Click **Configuration, Authentication, and then Unix**.
 - c. Set the Unix authentication attributes, as required for your deployment:
 - **Configuration Port:** Port that the *amunxd* daemon listens to at startup for configuration information. Default: 58946
 - **Authentication Port:** Port that the *amunxd* daemon listens for authentication requests. Default: 57946
 - **Timeout:** Minutes to complete the authentication. Default: 3
 - **Threads:** Number of simultaneous authentication sessions. Default: 5
 - **Authentication Level:** How much to trust an authentication mechanism. Default: 0
 - **PAM Service Name:** Configuration or stack that is shipped for the operating system. Default: other

Solaris systems: PAM Service Name=other

Linux systems: PAM Service Name=password

Linux Note: On some Linux systems, you might need to set PAM Service Name to a different value. If password fails, you will need to determine the PAM Service Name for your Linux system.

d. Click **Save and logout of the Console.**

2 **Login as superuser (root).**

3 **Start the amunxd daemon by running the amunxd script in the *zip-root/opensso/tools/helpers/bin* directory.**

For example:

```
# cd zip-root/opensso/tools/helpers/bin
# ./amunxd
```

Notes

- Run the amunxd daemon as root. If the daemon is started by a non-root user, Unix authentication will succeed only for NIS users. Local users in */etc/passwd* or */etc/shadow* on Solaris systems will not be able to authenticate.
- The Unix authentication service Configuration Port in the Administration Console and the port the amunxd process is started with (default 58946) must match. If you change the port in the Administration Console, use the *-c portnumber* option to start the amunxd process. For example:

If the Configuration Port is changed from the default value (58946) using the OpenSSO Enterprise Admin Console, run the amunxd script with the *-c* and *-p* arguments to specify the new port and IP address, respectively. For example:

```
# ./amunxd -c portnumber
```

- If the you want the amunxd process to accept connections from systems other than the localhost (that is, the OpenSSO Enterprise host), use the following options:

```
-i N -a ipaddr1 ... -a ipaddrN
```

where *N* is the number of IP addresses you want to specify, and *ipaddr1* ... "*ipaddrN*" are the IP addresses in the 3-dot (111.111.111.111) format of the systems that amunxd is to accept connections from.

Implementing OpenSSO Enterprise Session Failover

Sun™ OpenSSO Enterprise provides a web container independent session failover implementation using Sun Java System Message Queue (Message Queue) as the communications broker and the Oracle Berkeley DB as the session store database. This chapter describes these topics:

- [“Overview of OpenSSO Enterprise Session Failover” on page 53](#)
- [“Installing and Configuring the OpenSSO Enterprise Session Failover Components” on page 56](#)
- [“Configuring Session Failover in the OpenSSO Enterprise Console” on page 64](#)

Overview of OpenSSO Enterprise Session Failover

- [“OpenSSO Enterprise Session Failover Components” on page 53](#)
- [“OpenSSO Enterprise Session Failover Flow” on page 56](#)

OpenSSO Enterprise Session Failover Components

A OpenSSO Enterprise session failover deployment scenario includes these components:

- Two or more OpenSSO Enterprise instances running on different host servers and configured as a site.

To configure the OpenSSO Enterprise instances as a site, use one of these methods:

- When you run the Configurator for the OpenSSO Enterprise instances, specify the same Site Name and load balancer Primary URL on the Site Configuration page for each instance. For information, see [“Configuring OpenSSO Enterprise With a Custom Configuration” on page 34](#).

or

- If you did not configure the deployment as a site when you ran the Configurator, use either the Administrator Console or the `ssoadm` command-line utility to configure the OpenSSO Enterprise instances as a site.
- Load balancer for the OpenSSO Enterprise instances.
- Message Queue brokers, running in cluster mode on different servers.
- Oracle Berkeley DB client (`amsessiondb`) and database, running on the same servers as the Message Queue brokers.

OpenSSO Enterprise uses the Oracle Berkeley DB Java Edition as the session data store. For information see <http://www.oracle.com/database/berkeley-db/je/index.html>.

- Client requests, which can originate from a Web browser, C or Java application using the OpenSSO Enterprise SDK, or a J2EE or web policy agent.
- OpenSSO Enterprise configuration data store (not shown in the figure):
 - Sun Java System Directory Server: All OpenSSO Enterprise instances must access the same Directory Server.
 - or
 - OpenSSO data store: Instances must be configured for replication and act as a single directory server.

The configuration data store must be running and accessible in the deployment.

- OpenSSO Enterprise user data store (not shown in the figure):
 - Sun Java System Directory Server
 - Microsoft Active Directory
 - IBM Tivoli Directory Server

Note: The OpenSSO user data store is recommended only for prototype, proof of concept (POC), or developer deployments that have a small number of users. It is not recommended for production deployments.

The following figure shows a session failover deployment with three OpenSSO Enterprise instances. (The OpenSSO Enterprise configuration data store and user data store are not shown.)

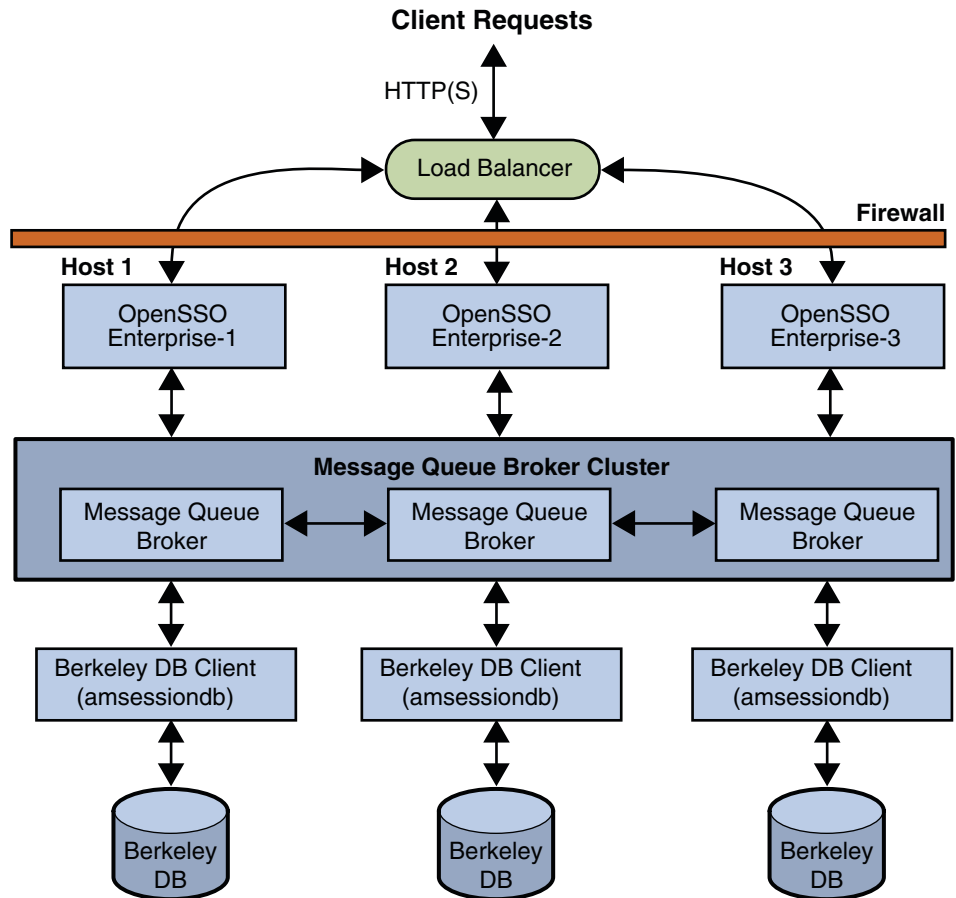


FIGURE 6-1 OpenSSO Enterprise Session Failover Components

OpenSSO Enterprise Session Failover Flow

OpenSSO Enterprise session failover follows the Message Queue publish/subscribe delivery model:

1. When a user initiates, updates, or ends a session, the OpenSSO Enterprise instance publishes a session creation, update, or deletion message to the Message Queue broker cluster.
2. The Oracle Berkeley DB client (`amsessiondb`) subscribes to the Message Queue broker cluster, reads the session messages, and stores the session operations in the database.

The OpenSSO Enterprise instances communicate with each other using an internal routing mechanism. If an OpenSSO Enterprise instance goes down due to a single hardware or software problem, a user's session associated with that instance moves to a secondary OpenSSO Enterprise instance, as follows:

1. The secondary OpenSSO Enterprise instance publishes a query request to the Message Queue broker cluster to get the user's session information.
2. The Oracle Berkeley DB client (`amsessiondb`) receives the query request, retrieves the corresponding user entry from the session database, and then publishes the user's session information to the Message Queue broker cluster.
3. The secondary OpenSSO Enterprise instance receives the response with the user's session information from the Message Queue broker and continues the session, without losing any session information or requiring the user to login again.

If a Message Queue broker goes down, OpenSSO Enterprise continues to operate in non-session failover mode. When the Message Queue broker is later restarted, OpenSSO Enterprise returns to session failover mode.

For more information about the Message Queue components and the publish/subscribe delivery model, see the *Sun Java System Message Queue 4.1 Technical Overview* in the following collection:

<http://docs.sun.com/coll/1307.3>

Installing and Configuring the OpenSSO Enterprise Session Failover Components

To install and configure the OpenSSO Enterprise session failover components, follow this procedure on each server in the Message Queue broker cluster:

- “Unzipping the `ssoSessionTools.zip` File” on page 57
- “Running the Session Failover setup Script” on page 58
- “Creating a New User to Connect to the Message Queue Broker (Optional)” on page 59

- “Editing the `amsessiondb` Script (if Needed)” on page 60
- “Encrypting the Message Queue Broker Password Using the `amsfopassword` Script (Required)” on page 60
- “Running the `amsfo` Script to Start and Stop the Session Failover Components” on page 61

Unzipping the `ssoSessionTools.zip` File

The `ssoSessionTools.zip` file, which is part of the `opensso.zip` file, contains the session failover scripts, JAR, properties, and related files.

▼ To Unzip the `ssoSessionTools.zip` File

Before You Begin Unzip the `opensso.zip` file. The `ssoSessionTools.zip` file is then available in the `zip-root/opensso/tools` directory, where *zip-root* is where you unzipped `opensso.zip`.

- 1 **Log in to the server where you want to install and configure the session failover components.**
- 2 **Create a new directory to unzip the `ssoSessionTools.zip`. For example: *sfo-zip-root***
- 3 **Copy the `ssoSessionTools.zip` file to the new directory.**
- 4 **Unzip the `ssoSessionTools.zip` file in the new directory.**

The following table describes the layout after you unzip the `ssoSessionTools.zip` file. The directory where you unzip `ssoSessionTools.zip` is represented by *sfo-zip-root*.

<i>sfo-zip-root</i> File or Directory	Description
<code>README.txt</code>	Description of the <code>ssoSessionTools.zip</code> file.
<code>setup</code>	Script to install the session tools on Solaris and Linux systems.
<code>setup.bat</code>	Script to install the session tools on Windows systems.
<code>ext</code> directory	<ul style="list-style-type: none"> ■ Message Queue JAR files for Solaris SPARC, Solaris x86, Linux, and Windows systems. ■ Oracle Berkeley DB JAR file (<code>je.jar</code>)
<code>lib</code> directory	<ul style="list-style-type: none"> ■ JAR file for the setup scripts (<code>am_session_setup.jar</code>) ■ JAR file for the session API (<code>am_sessiondb.jar</code>)
<code>locale</code> directory	Properties file for the session API (<code>amSessionDB.properties</code>)
<code>template</code> directory	Script templates for Solaris, Linux, and Windows systems.

Running the Session Failover setup Script

The session failover setup script installs these files:

- Sun Java System Message Queue JAR and related files
- Oracle Berkeley DB JAR and related files
- `amsfo`, `amsfopassword`, and `amsessiondb` scripts on Solaris and Linux system
`amsfo.pl`, `amsfopassword.bat`, and `amsessiondb.bat` on Windows systems
- `amsfo.conf` session failover configuration file

▼ To Run the Session Failover setup Script

Before You Begin

- The setup script requires Java Runtime Environment (JRE) 1.5 or later. Make sure that your `JAVA_HOME` environment variable points to your JDK installation directory.
- On Solaris and Linux systems, you might need to issue the following command before you run the setup script: `chmod +x setup`

- **In the directory where you unzipped the `ssoSessionTools.zip` file, run the setup script.**

On Solaris and Linux systems, use this syntax to run the setup script:

`setup -p|--path dirname`

where *dirname* is a directory under the current directory where the setup script places the session failover scripts and related files. If *dirname* does not exist, the script will create the directory for you.

For example: `# ./setup -p sfo`

Considerations:

- On Windows systems, run the `setup.bat` script.
- If you run the setup script without any options, the script prompts you for a path.
- If the path contains a space, run the setup script without any options and then provide the path when you are prompted.
- To display the help for the setup script: `setup -h|--help`

The setup (or `setup.bat`) script installs the session failover scripts and related files in the following directories:

<i>sfo</i> -zip-root Directory	Script or File
<code>jqmq/jmq</code>	Message Queue scripts and related files
<i>dirname</i> /config/lib	<code>amsfo.conf</code> session failover configuration file

<i>sfo-zip-root</i> Directory	Script or File
<i>dirname/bin</i>	<ul style="list-style-type: none"> ■ Scripts to start and stop the Message Queue broker and <code>amssessiondb</code> client: <ul style="list-style-type: none"> ■ <code>amsfo</code> on Solaris and Linux systems ■ <code>amsfo.pl</code> on Windows systems ■ Scripts to run the Oracle Berkeley DB client (called by <code>amsfo</code>): <ul style="list-style-type: none"> ■ <code>amssessiondb</code> on Solaris and Linux systems ■ <code>amssessiondb.bat</code> on Windows systems ■ Scripts to encrypt the Message Queue broker user password: <ul style="list-style-type: none"> ■ <code>amsfopassword</code> on Solaris and Linux systems ■ <code>amsfopassword.bat</code> on Windows systems

Creating a New User to Connect to the Message Queue Broker (Optional)

OpenSSO Enterprise requires a user and password to connect to the Message Queue broker. Message Queue provides the default guest user with password `guest` to connect to the broker.

However, if you don't want to use the guest user, you can create a new user and password using the Message Queue User Manager utility (`imqusermgr`).

The following example creates a new user named `openssomqusr` on a Solaris system.

▼ To Create a New User to Connect to the Message Queue Broker

- 1 **Change to the Message Queue `/bin` directory. For example:**

```
# cd sfo-zip-root/sfo/jmq/bin
```
- 2 **Create a new broker instance named `mqbroker`:**

```
# ./imqbrokerd -name mqbroker -port 7777 &
```
- 3 **Check to make sure that the new broker instance is running:**

```
netstat -an | grep 7777
```
- 4 **Create the new Message Queue user and password:**

```
# ./imqusermgr add -u openssomqusr -p mq-password -i mqbroker
```
- 5 **Disable the guest user so guest will not be able to access OpenSSO Enterprise. For example:**
 - a.

```
# ./imqusermgr update -u guest -a false -i mqbroker
```

b. When prompted, reply y to update the guest user:

Are you sure you want to update user guest? (y/n) [n] y

6 Stop the broker instance. For example:

a. # ./imqcmd shutdown bkr -b mq1.example.com:7777 -u admin

b. When prompted, enter the admin password.

See Also For more information, see [“Using the User Manager Utility” in Sun Java System Message Queue 4.1 Administration Guide.](#)

Editing the amsessiondb Script (if Needed)

The `amsfo` script calls `amsessiondb` to start the Oracle Berkeley DB client (`amsessiondb`), create the database, and set specific database values. The `amsessiondb` script contains variables that specify various default paths and directories. For example:

```
JAVA_HOME=/usr/jdk/entsys-j2se/
AM_HOME=/opensso/tools/sfo-zip-root/sfo
JMS_JAR_PATH=/usr/share/lib
IMQ_JAR_PATH=/usr/share/lib
BDB_JAR_PATH=/usr/share/db.jar
BDB_SO_PATH=/usr/lib
```

If any of these components are not installed in the directories shown in the `amsessiondb` script, edit the script and set each variable, as needed, to the path where the component is installed.

Encrypting the Message Queue Broker Password Using the amsfopassword Script (Required)

The `amsfopassword` script accepts the Message Queue broker password in clear text and returns the encrypted password in a file. You can then use this file as input to the `amsfo` script by setting the `PASSWORDFILE` variable in the `amsfo.conf` configuration file.

To run the `amsfopassword` script, use the following syntax:

```
amsfopassword
-f|--passwordfile password-file -e|--encrypt clear-text-password
```

- *password-file* is the path to the destination file where `amsfopassword` stores the encrypted password.
- *clear-text-password* is the clear text password that `amsfopassword` encrypts.

To display help, specify `-h` | `--help`.

▼ To Encrypt the Message Queue Broker Password Using the `amsfopassword` Script

- 1 **On the server where you ran the setup script, run the `amsfopassword` script.**

For example, on a Solaris system:

```
# cd /sfo-zip-root/sfo/bin
# ./amsfopassword -f /sfo-zip-root/sfo/mqpassword -e cleat-text-password
```

You are not required to run `amsfopassword` as superuser (`root`).

- 2 **Use the encrypted password in the `mqpassword` file as input to the `amsfo` script by setting the `PASSWORDFILE` variable in the `amsfo.conf` file.**

For information about the `PASSWORDFILE` variable, see [Table 6–1](#).

Running the `amsfo` Script to Start and Stop the Session Failover Components

The `amsfo` script (or `amsfo.pl` on Windows systems) reads variables in the `amsfo.conf` configuration file and then performs these functions:

- Starts or stops the Message Queue broker and the Oracle Berkeley DB client (`amsessiondb`) on each server in the broker list (`CLUSTER_LIST` variable).
- Deletes and then recreates the Oracle Berkeley DB database, if requested.
- Writes the `amsessiondb.log`, `jmql.pid`, and `amdb.pid` files in the `/tmp/amsession/logs/` directory. The default log directory is determined by the `LOG_DIR` variable in the `amsfo.conf` file.

To run the script on Windows systems, Active Perl version 5.8 or later is required.

To run `amsfo`, use the following syntax:

```
amsfo start | stop
```

The `amsfo` command then automatically finds the `amsfo.conf` file.

The following table describes the variables in the `amsfo.conf` file. Some variables are set when you run the `setup` (or `setup.pl`) script. Before you run the `amsfo` script, set other variables as required for your deployment.

TABLE 6-1 amsfo.conf Configuration File Parameters

Variable	Description
AM_HOME_DIR	<p>Specifies the following directory: <i>sfo-zip-root/dirname</i></p> <p>where:</p> <ul style="list-style-type: none"> ■ <i>sfo-zip-root</i> is where you unzipped the <i>ssoSessionTools.zip</i> file. ■ <i>dirname</i> is the name you specified when you ran the setup script to install the session failover scripts and related files.
AM_SFO_RESTART	<p>Specifies (<i>true</i> or <i>false</i>) whether the script should automatically restart the Oracle Berkeley DB client (<i>amsessiondb</i>).</p> <p>The default is <i>true</i> (restart the <i>amsessiondb</i> client).</p>
CLUSTER_LIST	<p>Specifies the Message Queue broker list participating in the cluster. The format is:</p> <p><i>host1:port,host2:port, ... hostn:port</i></p> <p>For example:</p> <p><i>mq1.example.com:7777,mq2.example.com:7777, mq3.example.com:7777</i></p> <p>You can deploy the Message Queue brokers on the same servers that are running OpenSSO Enterprise instances. However, for improved performance, consider installing the brokers on different servers.</p>
DATABASE_DIR	<p>Specifies the directory where the session database files will be created.</p> <p>Default: <i>/tmp/amsession/sessiondb</i></p>
DELETE_DATABASE	<p>Specifies (<i>true</i> or <i>false</i>) whether the script should delete and then create a new database each time the Oracle Berkeley DB client (<i>amsessiondb</i>) is restarted.</p> <p>Default: <i>true</i></p>
LOG_DIR	<p>Specifies the location of the log directory.</p> <p>Default: <i>/tmp/amsession/logs</i></p>
START_BROKER	<p>Specifies (<i>true</i> or <i>false</i>) whether the Message Queue broker should be started with the <i>amsessiondb</i> process on the same server:</p> <p><i>true</i> - The Message Queue broker will run on the same server as the <i>amsessiondb</i> process.</p> <p><i>false</i> - The Message Queue broker and the <i>amsessiondb</i> process will run on different servers.</p> <p>Default: <i>true</i></p>

TABLE 6-1 `amsfo.conf` Configuration File Parameters (Continued)

Variable	Description
BROKER_INSTANCE_NAME	Specifies the name of the Message Queue broker instance to start. For example: mqbroker
BROKER_PORT	Specifies the port for the local Message Queue broker instance. Default: 7777
BROKER_VM_ARGS	Specifies the Java VM arguments. Set to a maximum of 1024m, based on the system resources. Default: "-Xms256m -Xmx512m"
USER_NAME	Specifies the user name used to connect to the Message Queue broker. Default: guest If you specified a different user name, as described in “Creating a New User to Connect to the Message Queue Broker (Optional)” on page 59, set USER_NAME to that name.
PASSWORDFILE	Location of the password file that contains the encrypted password used to connect to the Message Queue broker. To generate the encrypted password, use the <code>amsfopassword</code> script, as described in “Encrypting the Message Queue Broker Password Using the <code>amsfopassword</code> Script (Required)” on page 60. Default: <code>sfo-zip-root/dirname/.password</code>
AMSESSIONDB_ARGS	<code>amsessiondb</code> script arguments. The <code>amsessiondb</code> script is called by the <code>amsfo</code> (or <code>amsfo.pl</code>) script. To determine the list of arguments, run: <code>amsessiondb -h</code>
lbServerPort	Specifies the port for the load balancer. Default: 80
lbServerProtocol	Specifies the protocol (http or https) used to access the load balancer. The default is http.
lbServerHost	Specifies the name of the load balancer. For example: <code>lbhost.example.com</code>
SiteID	Specifies the identifier for the new site (and the load balancer). SiteID can be any value greater than the Server IDs that already exist in the platform server list. Default: 10

▼ To Run the `amsfo` Script

Before You Begin Stop each of the OpenSSO Enterprise instances in the session failover deployment.

- 1 **Set the variables in the `amsfo.conf` file, as required for your deployment.**

For a description of all variables, see [Table 6–1](#).

- 2 **Run the `amsfo` script on Solaris or Linux systems or the `amsfo.pl` script on Windows systems.**

For example, to start the session failover components on a Solaris system:

```
# cd /sfo-zip-root/sfo/bin
# ./amsfo start
```

The `amsfo` command then automatically finds the `amsfo.conf` file and displays status information as it runs.

- 3 **To check the results, see the `/var/tmp/amsfo.log` file.**

Configuring Session Failover in the OpenSSO Enterprise Console

▼ To Configure Session Failover in the OpenSSO Enterprise Console

Before You Begin If necessary, start each OpenSSO Enterprise instance in the session failover deployment.

- 1 **Log in to the OpenSSO Enterprise Console as `amadmin`.**
- 2 **Click Configuration, Global, and then Session.**
- 3 **Under Secondary Configuration Instance, click the site Name for the session failover configuration.**
- 4 **On the Edit Sub Configuration page, specify the Global Attributes.**

When applicable, use the same values for the corresponding parameters in the `amsfo.conf` configuration file.

- **Session Store User** is the user that connects to the Message Queue broker. For example: `openssomquser`
- **Session Store Password** (and confirmation) is the password for the user that connects to the Message Queue broker.
- **Maximum Wait Time** should be the default value of 5000 milliseconds.

- **Database Url** is the Message Queue broker address list, which is the list of Message Queue brokers participating in the cluster. For example:
`mq1.example.com:7777,mq2.example.com:7777,mq3.example.com:7777`
- **Session Failover Enabled** must be Enabled.

5 Check Save and log out of the console.

6 Restart each OpenSSO Enterprise instance in the site for the new session failover values to take effect.

Deploying a Distributed Authentication UI Server

A Sun™ OpenSSO Enterprise Distributed Authentication UI server provides for secure, distributed authentication across two firewalls in an OpenSSO Enterprise deployment.

A Distributed Authentication UI server does not run OpenSSO Enterprise. This server exists only to provide the customizable authentication interface between end users and an OpenSSO Enterprise instance.

Topics in this chapter include:

- “Distributed Authentication UI Server Overview” on page 67
- “Generating a Distributed Authentication UI Server WAR File” on page 70
- “Deploying the Distributed Authentication UI Server WAR File” on page 71
- “Configuring the Distributed Authentication UI Server” on page 71
- “Accessing the Distributed Authentication User Interface Web Application” on page 73

Distributed Authentication UI Server Overview

- “Distributed Authentication UI Server Deployment Scenario” on page 67
- “Requirements for a Distributed Authentication UI Server Deployment” on page 68
- “Flow for a Distributed Authentication End-User Request” on page 69

Distributed Authentication UI Server Deployment Scenario

You install the Distributed Authentication UI server subcomponent on one or more servers within the DMZ layer of an OpenSSO Enterprise deployment. This subcomponent acts as an authentication interface between end users and the OpenSSO Enterprise instances behind the second firewall, thus eliminating the exposure of the OpenSSO Enterprise service URLs to the end users.

The following figure shows a Distributed Authentication UI server deployment scenario.

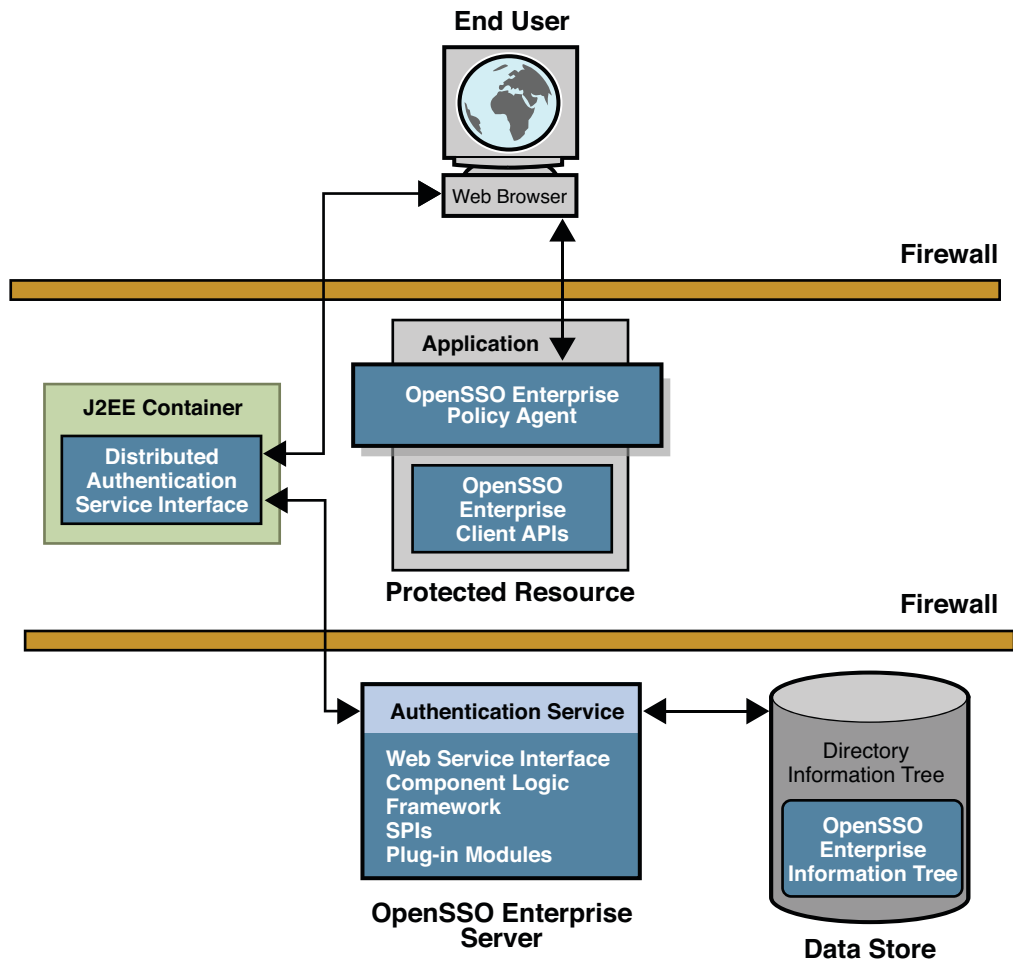


FIGURE 7-1 Distributed Authentication UI Server Deployment Scenario

Requirements for a Distributed Authentication UI Server Deployment

The Distributed Authentication UI server must be installed in a supported web container, as listed in [“OpenSSO Enterprise 8.0 Requirements” on page 18](#).

To generate a Distributed Authentication UI server WAR file, your `JAVA_HOME` environment variable must point to a JDK of version 1.5 or later.

Several other considerations for a Distributed Authentication UI server include:

- If you are deploying multiple Distributed Authentication UI servers behind a load balancer, stickiness is not required for the load balancer to talk to only one Distributed Authentication UI server for authentication process completion.
- The HTTP Basic and MSISDN authentication modules are not supported through the Distributed Authentication UI.

Flow for a Distributed Authentication End-User Request

In a typical deployment using one or more Distributed Authentication UI servers, an end-user request follows this flow:

1. An end user sends an HTTP or HTTPS request from a Web browser to access a protected resource.
2. If the request does not have a cookie containing an SSO token, the OpenSSO Enterprise policy agent issues a redirect to its authentication URL, which is the URL of the Distributed Authentication UI server in the DMZ (usually through a load balancer).
3. The end user follows the redirect and sends the request to the Distributed Authentication UI server.
4. The Distributed Authentication UI server sends the request to an OpenSSO Enterprise instance behind the second firewall to determine the appropriate authentication mechanism.
5. The OpenSSO Enterprise instance determines the appropriate authentication mechanism and then returns the authentication framework to the Distributed Authentication UI server.
6. Using the information from the OpenSSO Enterprise instance, the Distributed Authentication UI server returns a login page to the user's Web browser.
7. The end user replies with the login credentials (such as user name and password) to the Distributed Authentication UI server.
8. The Distributed Authentication UI server uses the OpenSSO Enterprise Client SDK to send the end user's credentials to the OpenSSO Enterprise instance behind the second firewall.
9. OpenSSO Enterprise tries to authenticate the end user using the appropriate authentication method:
 - If the authentication is successful, OpenSSO Enterprise returns the SSO token. The Distributed Authentication UI server sets the session cookie in the browser and then redirects the end user to the protected resource.

- If the authentication is not successful, OpenSSO Enterprise returns the appropriate error information.

Generating a Distributed Authentication UI Server WAR File

To generate a Distributed Authentication UI server WAR file, use the `jar` command to extract the files from the `opensso.war` file and then to generate the specialized WAR file.

▼ To Generate a Distributed Authentication UI Server WAR File

Before You Begin If you have not already done so, download and unzip the `opensso.zip` file. You will then need the following files:

- `zip-root/deployable-war/opensso.war` is the OpenSSO Enterprise WAR file that contains all components, including the Distributed Authentication UI server files.
- `zip-root/deployable-war/fam-distauth.list` specifies the files that are required to generate a Distributed Authentication UI server WAR file.
- `zip-root/deployable-war/distauth` contains the additional files you will need to deploy and configure a Distributed Authentication UI server.

where `zip-root` is the directory where you unzipped the `opensso.zip` file.

For more information about the `opensso.war` file, see [“Downloading OpenSSO Enterprise” on page 23](#).

- 1 **Make sure that your `JAVA_HOME` environment variable points to a JDK of version 1.5 or later.**
- 2 **Create a new staging directory and extract the files from `opensso.war` in this staging directory. For example:**

```
# mkdir dastaging
# cd dastaging
# jar xvf zip-root/opensso/deployable-war/opensso.war
```

- 3 **Create the Distributed Authentication UI server WAR using the files in `fam-distauth.list`:**

```
# cd dastaging
# jar cvf zip-root/opensso/deployable-war/openssoDistauth.war \
  @zip-root/opensso/deployable-war/fam-distauth.list
```

where `openssoDistauth.war` is the name of the new Distributed Authentication UI server WAR file.

Note: Some web containers require the Distributed Authentication WAR file name to use the same name as the deployment URI.

- 4 **Update the WAR file created in previous step with the additional files required for the Distributed Authentication UI server WAR. For example:**

```
# cd zip-root/opensso/deployable-war/distauth
# jar uvf zip-root/opensso/deployable-war/openssoDistauth.war *
```

You are now ready to configure the new `openssoDistauth.war`, as described in the next section.

Deploying the Distributed Authentication UI Server WAR File

▼ To Deploy the Distributed Authentication UI Server WAR File

- Before You Begin**
- The web container that you plan to use for the Distributed Authentication UI server must be installed. See [“Requirements for a Distributed Authentication UI Server Deployment” on page 68](#) for a list of the supported web containers.
 - One or more OpenSSO Enterprise full server instances must be running remotely in the deployment.
- 1 **Login as a user who has the following privileges:**
 - Access to the web container administration console, if you plan to deploy Distributed Authentication UI server WAR file using the console.
 - or
 - The capability to execute the web container's deploy command-line utility, if you plan to deploy the WAR file using the CLI.
 - 2 **Make sure that the Distributed Authentication UI server web container is running.**
 - 3 **Deploy the Distributed Authentication UI WAR file using the using the web container administration console or deployment command.**

Configuring the Distributed Authentication UI Server

OpenSSO Enterprise includes the Distributed Authentication UI server Configurator (`distAuthConfigurator.jsp`) to configure a Distributed Authentication UI server after you deploy the WAR file.

▼ To Configure the Distributed Authentication UI Server

- 1 **Make sure that the Distributed Authentication UI server web container is running.**

- 2 **Launch the Distributed Authentication UI server WAR file using the following URL:**

protocol://host.domain:port/distauth_uri

For example: `http://distauth.example.com:8080/openssoDistauth`

If the Distributed Authentication UI server is not already configured, you will be directed to the Configurator (`distAuthConfigurator.jsp`) page. (If the Distributed Authentication UI server is already configured, you will be directed to the login page.)

- 3 **On the Configurator page, specify the following information:**

- **Server Protocol** is the OpenSSO Enterprise server protocol: `http` or `https`. Default: `http`
Note: If the Distributed Authentication UI Server is being configured to use an SSL-enabled OpenSSO Enterprise server, you must import the root CA certificate for the server certificate on the OpenSSO Enterprise server into the trust store of the web container JVM on which the Distributed Authentication UI Server is being deployed. After you import the certificate, restart the web container instance.
- **Server Host** is the fully qualified host name of the system where OpenSSO Enterprise server is deployed.
- **Server Port** is the OpenSSO Enterprise server port number. Default: `8080`
- **Server Deployment URI** is the URI prefix for accessing the HTML pages, classes, and JAR files associated with OpenSSO Enterprise server.
- **DistAuth Server Protocol** is the protocol (`http` or `https`) used by the Distributed Authentication UI server web container. Default: `http`
- **DistAuth Server Host** is the fully qualified host name where the Distributed Authentication UI server is deployed.
- **DistAuth Server Port** is the port number on DistAuth Server Host where the Distributed Authentication UI server is deployed. Default: `80`
- **DistAuth Server Deployment URI** is the deployment URI that will be used on the host by the Distributed Authentication UI server.
- **DistAuth Cookie Name** is the cookie name used on the host by the Distributed Authentication UI server.
- **Debug directory** is the directory where the debug files will be created.
- **Debug level** is the level for the debug service. Values can be: `error`, `warning`, `message` or `off`. Default: `error`
- **Encryption Key** is the password encryption key.

- **Application user name** is the user name for the Distributed Authentication UI server application. For example: `UrlAccessAgent`
 - **Application user password** is the password of the user for the application.
 - **Confirm Application user password** is confirmation for the password.
- 4 After you have specified all configuration values (or accepted the default values), click **Configure**.
(Or, to reset all values, click **Reset**.)

Next Steps After the configuration finishes, you will get a message showing the location of the `AMDistAuthConfig.properties` configuration file. This file is created in the home directory of the runtime user who owns the web container instance on which the Distributed Authentication UI WAR file is deployed.

Important: It is highly recommended that you change the permissions of this configuration file to limit access to the sensitive configuration information.

Accessing the Distributed Authentication User Interface Web Application

To access the Distributed Authentication UI server application, use the following URL in your browser:

daserver_protocol://daserver_host:daserver_port/dadeploy_uri/UI/Login

Where:

- *daserver_protocol* is the protocol (`http` or `https`) used by the Distributed Authentication UI server web container instance.
- *daserver_host* is the fully qualified host name of the Distributed Authentication UI server.
- *daserver_port* is the port for the Distributed Authentication UI server host.
- *dadeploy_URI* is the deployment URI prefix for the Distributed Authentication UI server. The default value is the URI used to access the Configurator..

For example:

`https://daserver.example.com:80/openssoDistauth/UI/Login`

Note –

- In a production environment, the Distributed Authentication UI server web application is usually deployed in the DMZ layer. So, always specify the successful redirect URL to an absolute URL. For example:
`https://daserver.example.com:80/openssoDistauth/UI/Login?
goto=/absolute-successful-redirect-URL/`
 - For testing purposes, if you use the server returned default successful redirect URL (which is the server OpenSSO Enterprise Admin Console URL) , make sure that you change this URL from its relative value to the absolute value before your move to a production environment by using the server Administration Console (Authentication Configuration > Properties).
-

Deploying the Identity Provider (IDP) Discovery Service

Sun OpenSSO Enterprise 8.0 implements the Identity Provider Discovery profile (part of the SAMLv2 binding profiles) for its Identity Provider Discovery Service to keep track of the identity providers for each user. Deploying the IPP Discovery Service includes these steps:

- “Generating an IDP Discovery Service WAR File” on page 75
- “Configuring the IDP Discovery Service” on page 76

Generating an IDP Discovery Service WAR File

To generate an IDP Discovery Service WAR file, use the `jar` command to extract the files from the `opensso.war` file and then to generate the specialized WAR file.

▼ To Generate an IDP Discovery Service WAR File

Before You Begin Download and unzip the `opensso.zip` file. You will then need the following files:

- `zip-root/deployable-war/opensso.war` is the OpenSSO Enterprise WAR file that contains all components, including the IDP Discovery Service files.
- `zip-root/deployable-war/fam-idpdiscovery.list` specifies the files that are required to generate an IDP Discovery Service WAR file.
- `zip-root/deployable-war/idpdiscovery` directory contains additional files you will need to deploy and configure the IDP Discovery Service.

where `zip-root` is where you unzipped the `opensso.zip` file.

For more information about the `opensso.war` file, see “[Downloading OpenSSO Enterprise](#)” on [page 23](#).

- 1 **Make sure that your `JAVA_HOME` environment variable points to JDK 1.5 or later.**

- 2 **Create a new staging directory and extract the files from `opensso.war` in this staging directory. For example:**

```
# mkdir idpdiscovery
# cd idpdiscovery
# jar xvf zip-root/opensso/deployable-war/opensso.war
```

- 3 **Create the IDP Discovery Service WAR using the files in `fam-idpdiscovery.list`:**

```
# cd idpdiscovery
# jar cvf zip-root/opensso/deployable-war/idpdiscovery.war \
@zip-root/opensso/deployable-war/fam-idpdiscovery.list
```

where `idpdiscovery.war` is the name of the new IDP Discovery Service WAR file.

- 4 **Update the `idpdiscovery.war` file created in previous step with the additional files required for the IDP Discovery Service. For example:**

```
# cd zip-root/opensso/deployable-war/idpdiscovery
# jar uvf zip-root/opensso/deployable-war/idpdiscovery.war *
```

You are now ready to configure the new `idpdiscovery.war`, as described in the next section.

Configuring the IDP Discovery Service

OpenSSO Enterprise includes the IDP Discovery Service Configurator (`Configurator.jsp`) to configure the service.

▼ To Configure the IDP Discovery Service

- 1 **Login as a user who has the following privileges:**
 - Access to the web container administration console, if you plan to deploy `idpdiscovery.war` using this console.
 - or
 - The capability to execute the web container's deploy command-line utility, if you plan to deploy `idpdiscovery.war` using the CLI.
- 2 **Deploy the `idpdiscovery.war` to the web container using either the web container administration console or CLI command.**
- 3 **Launch the Configurator using the following URL:**

protocol://host.domain:port/idpdiscovery

For example: `http://idpdiscoveryhost.example.com:8080/idpdiscovery`

If the IDP Discovery Service is not already configured, you will be directed to the Configurator page.

4 On the Configurator page, specify the following information:

- Debug Directory:
- Debug Level: error (default), warning, message, or off.
- Cookie Type: PERSISTENT (default) or SESSION
- Cookie Domain:
- Secure Cookie: True or False (default)
- Encode Cookie: True (default) or False

5 Click Configure.

6 On the SP host machine, use the console to create a Circle of Trust with the IDP Discovery Service URL used as the prefix for the value of the Reader and Writer URL attributes. For example:

SAML2 Writer Service URL:

`http://idp-discovery-server-machine:port/idpdiscovery/saml2writer`

SAML2 Reader Service URL:

`http://idp-discovery-server-machine:port/idpdiscovery/saml2reader`

7 On the IDP host machine, use the console to create a Circle of Trust with the value of the prefix attribute also set to the identity provider discovery service URL. For example:

`http://idp-discovery-server-machine:port/idpdiscovery`

8 Generate metadata for both the IDP and the SP using the `ssoadm` command-line utility with the `create-metadata-templ` option.

9 Load the SP metadata into the IDP machine.

10 Change the value of the host in the IDP metadata from 0 or remote.

11 Load the IDP metadata into the SP machine.

After this configuration, the values of the Writer URL and Reader URL in each Circle of Trust are the URL of the IDP Discovery Service.

Next Steps Perform the SAMLv2 test cases for SP-initiated and IDP-initiated single sign-on and single logout. Each time you perform these operations from the SP side, the Discovery Service logs will show the redirection to the IDP.

Installing the OpenSSO Enterprise Console Only

This chapter describes how to install only the Sun™ OpenSSO Enterprise Administration Console, including:

- [“Requirements to Deploy Only the Console” on page 79](#)
- [“Generating a Console Only WAR File” on page 79](#)
- [“Deploying and Configuring the Console Only WAR File” on page 80](#)
- [“Accessing the Console” on page 82](#)

Requirements to Deploy Only the Console

To deploy only the Administration Console, your deployment must meet the following requirements:

- You must deploy the Console to a supported web container, as listed in the [“OpenSSO Enterprise 8.0 Requirements” on page 18](#).
- One or more OpenSSO Enterprise full server instances must be running remotely in the deployment.
- If you currently have a console only deployment, you must first uninstall the console. See [“Uninstalling an OpenSSO Enterprise Console Only Deployment” on page 124](#).

Generating a Console Only WAR File

To generate a console only WAR file, use the `jar` command to extract the files from the `opensso.war` file and then to generate the specialized WAR file.

▼ To Generate a Console Only WAR File

Before You Begin Download and unzip the `opensso.zip` file. You will then need the following files:

- `zip-root/deployable-war/opensso.war` is the OpenSSO Enterprise WAR file that contains all components, including the console files.
- `zip-root/deployable-war/fam-console.list` specifies the files that are required to generate a console only WAR file.
- `zip-root/deployable-war/console` contains additional files you will need to deploy and configure the console.

where `zip-root` is where you unzipped the `opensso.zip` file.

For more information about the `opensso.war` file, see [“Downloading OpenSSO Enterprise” on page 23](#).

- 1 **Make sure that your `JAVA_HOME` environment variable points to JDK 1.5 or later.**
- 2 **Create a new staging directory and extract the files from `opensso.war` in this staging directory. For example:**

```
# mkdir consolestaging
# cd consolestaging
# jar xvf zip-root/opensso/deployable-war/opensso.war
```

- 3 **Create the Console only WAR using the files in `fam-console.list`:**

```
# cd consolestaging
# jar cvf zip-root/opensso/deployable-war/consoleonly.war \
    @zip-root/opensso/deployable-war/fam-console.list
```

where `consoleonly.war` is the name of the new Console only WAR file.

- 4 **Update the WAR file created in previous step with the additional files required for the specific Console only WAR. For example:**

```
# cd zip-root/opensso/deployable-war/console
# jar uvf zip-root/opensso/deployable-war/consoleonly.war *
```

You are now ready to configure the new `consoleonly.war`, as described in the next section.

Deploying and Configuring the Console Only WAR File

OpenSSO Enterprise includes the Console only WAR File Configurator (`Configurator.jsp`) to configure a Console only WAR file.

▼ To Deploy and Configure the Console Only WAR File

- 1 **Login as a user who has the following privileges:**

- Access to the web container administration console, if you plan to deploy `consoleonly.war` using this console.
- or
- The capability to execute the web container's deploy command-line utility, if you plan to deploy `consoleonly.war` using the CLI.

2 Deploy `consoleonly.war` using either the web container administration console or CLI.

3 Launch the Configurator using the following URL:

protocol://host.domain:port/console

For example: `http://host.example.com:8080/console`

If the Console only deployment is not already configured, you will be directed to the Configurator page. (If the deployment is already configured, you will be directed to the login page.)

4 On the Configurator page, specify the following information:

- **Server Protocol** is the OpenSSO Enterprise server protocol: `http` or `https`. Default: `http`
- **Server Host** is the fully qualified host name of the system where OpenSSO Enterprise server is deployed.
- **Server Port** is the OpenSSO Enterprise server port number. Default: `58080`
- **Server Deployment URI** is the URI prefix for accessing the HTML pages, classes, and JAR files associated with OpenSSO Enterprise server.
Important: This value must include the leading slash (/).
- **Application user name** is the user name for the Console only application.
- **Application user password** is the password of the user for the application.
- **Administration Console Protocol** is the protocol (`http` or `https`) used by the Console only server web container. Default: `http`
- **Administration Console Host** is the fully qualified host name where the Console only server is deployed.
- **Administration Console Port** is the port number for the Console only server is deployed.
- **Administration Console Deployment URI** is the deployment URI Console only server. Default: `/console`
- **Administration Console Debug directory** is the directory where the debug files will be created.

5 After you have specified all configuration values (or accepted the default values), click **Configure.**

(Or, to reset all values, click **Reset**.)

Next Steps After the configuration finishes, you will get a message showing the location of the Console only configuration file. This file is created in the home directory of the runtime user who owns the web container instance on which Console only WAR file is deployed.

Important: It is highly recommended that you change the permissions of this configuration file to limit access to the sensitive configuration information.

Accessing the Console

To access the Console in a Console only deployment, use the following URL in your browser:

consoleonly_protocol://consoleonly_host:consoleonly_port/consoleonly_uri

Where:

- *consoleonly_protocol* is the protocol (http or https) used by the Console only server web container instance.
- *consoleonly_host* is the fully qualified host name of the Console only server.
- *consoleonly_port* is the port for the Console only server host.
- *consoleonly_uri* is the deployment URI prefix for the Console only server. The default value is /console.

For example:

`http://openssoconsole.example.com:58080/console`

Installing OpenSSO Enterprise Server Only

In some deployments, you might need to install the Sun™ OpenSSO Enterprise server without the administration console. For instance, you might want to use only the command-line utilities such as `ssoadm` to access the server. This chapter describes these topics:

- “Requirements to Deploy OpenSSO Enterprise Server Only” on page 83
- “Generating a WAR File to Deploy OpenSSO Enterprise Server Only” on page 83
- “Deploying OpenSSO Enterprise Server Only” on page 84

Requirements to Deploy OpenSSO Enterprise Server Only

You must deploy the OpenSSO Enterprise server to a supported web container, as listed in the “OpenSSO Enterprise 8.0 Requirements” on page 18.

Generating a WAR File to Deploy OpenSSO Enterprise Server Only

To generate a WAR file to deploy OpenSSO Enterprise server without an administration console, use the `jar` command to extract the files from the `opensso.war` file and then to generate the specialized WAR file.

▼ To Generate a WAR File to Deploy OpenSSO Enterprise Server Only

Before You Begin Download and unzip the `opensso.zip` file. You will then need the following files:

- `zip-root/deployable-war/opensso.war` is the OpenSSO Enterprise WAR file that contains all components, including the server only files.

- `zip-root/deployable-war/fam-noconsole.list` specifies the files that are required to generate a server only WAR file.
- `zip-root/deployable-war/noconsole` contains additional files you will need to deploy the server only.

where `zip-root` is where you unzipped the `opensso.zip` file.

For more information about the `opensso.war` file, see [“Downloading OpenSSO Enterprise” on page 23](#).

- 1 **Make sure that your `JAVA_HOME` environment variable points to JDK 1.5 or later.**
- 2 **Create a new staging directory and extract the files from `opensso.war` in this staging directory. For example:**

```
# mkdir noconsolestaging
# cd noconsolestaging
# jar xvf zip-root/opensso/deployable-war/opensso.war
```

- 3 **Create the server only WAR using the files in `fam-noconsole.list`:**

```
# cd noconsolestaging
# jar cvf zip-root/opensso/deployable-war/noconsole.war \
  @zip-root/opensso/deployable-war/fam-noconsole.list
```

where `noconsole.war` is the name of the new server only WAR file.

- 4 **Update the WAR file created in previous step with the additional files required for the specific server only WAR. For example:**

```
# cd zip-root/opensso/deployable-war/noconsole
# jar uvf zip-root/opensso/deployable-war/noconsole.war *
```

You are now ready to configure the new `noconsole.war`, as described in the next section.

Deploying OpenSSO Enterprise Server Only

▼ To Deploy OpenSSO Enterprise Server Only

- 1 **Login as a user who has the following privileges:**
 - Access to the web container administration console, if you plan to deploy Distributed Authentication UI server WAR file using the console.
 - or
 - The capability to execute the web container's deploy command-line utility, if you plan to deploy the WAR file using the CLI.

- 2 Make sure that the web container for the server only deployment is running.
- 3 Deploy the server only WAR file using the using the web container administration console or deployment command.
- 4 Restart the OpenSSO Enterprise Server web container.

Next Steps Configure the server only deployment using the Configurator:

- [Chapter 3, “Configuring OpenSSO Enterprise Using the GUI Configurator”](#)
- [Chapter 4, “Configuring OpenSSO Enterprise Using the Command-Line Configurator”](#)

Installing the OpenSSO Enterprise Client SDK

The Sun™ OpenSSO Enterprise Client SDK is a smaller version of the OpenSSO Enterprise SDK that includes only the client-side Java classes and configuration properties. You can use the Client SDK to write remote standalone or web applications that access an OpenSSO Enterprise server to use services such as authentication, SSO, authorization, auditing, logging, and the Security Assertion Markup Language (SAML).

The Client SDK also includes sample applications that you can deploy to help you write your own custom applications.

This chapter describes:

- [“OpenSSO Enterprise Client SDK Requirements” on page 87](#)
- [“Installing the OpenSSO Enterprise Client SDK” on page 88](#)
- [“Compiling and Running the Client SDK Samples” on page 89](#)

OpenSSO Enterprise Client SDK Requirements

The requirements to use the Client SDK include:

- OpenSSO Enterprise server must be running on a remote server. You will need the following information about this remote installation:
 - Protocol (http or https) used by web container instance on which the OpenSSO Enterprise server is deployed.
 - Fully qualified domain name (FQDN) of the host on which the OpenSSO Enterprise server is deployed.
 - Port on which the OpenSSO Enterprise server is running.
 - Deployment URI for the OpenSSO Enterprise server (default is opensso).
 - Default Agent user (UrlAccessAgent) password that you entered when you ran the OpenSSO Enterprise Configurator.

- If you are writing a web application, you need a web container supported by OpenSSO Enterprise. For the list of supported web containers, see the “[OpenSSO Enterprise 8.0 Requirements](#)” on page 18.

Installing the OpenSSO Enterprise Client SDK

▼ To Install the OpenSSO Enterprise Client SDK

Before You Begin

- If you have not already done so, download and unzip the opensso.zip file, as described in “[Downloading OpenSSO Enterprise](#)” on page 23.
The Client SDK and samples are then available in the *zip-root/opensso/samples/opensso-client.zip* file, where *zip-root* is the directory where you unzipped opensso.war.
- If you plan to install the Client SDK in a web container, the web container must be installed on the server where you plan to deploy the Client SDK.

- 1 **On the server where you plan to deploy the Client SDK, copy the opensso-client.zip to a staging directory.**
- 2 **In the directory from Step 1, unzip the opensso-client.zip file.**

The following table describes the layout after you unzip the opensso-client.zip file. The directory where you unzip the file is represented by *opensso-client-zip-root*.

<i>opensso-client-zip-root</i> Directory	Description
/sdk	Client SDK CLI-based samples, which you can run in a standalone JVM outside of a web container: <ul style="list-style-type: none">■ /source contains the source files that require compilation.■ /scripts contains the scripts to compile and run the samples.■ /resources contains the properties files required to run the samples.■ /lib contains the JAR files required by the Client SDK.■ /classes contains the compiled classes from the source files.
/war	Client SDK WAR files, which include the web-based client samples: <ul style="list-style-type: none">■ opensso-client-jdk15.war is for web containers running JDK 1.5 or later■ opensso-client-jdk14.war is for web containers running JDK 1.4.x <p>Deploy these files using the web container administration console or command-line utility.</p>

Compiling and Running the Client SDK Samples

▼ To Compile and Run the Client SDK Samples

Before You Begin If you have not already do so, unzip the `opensso-client.zip` file, as described in [“Installing the OpenSSO Enterprise Client SDK” on page 88](#).

The Client SDK samples are then available in the `opensso-client-zip-root/sdk/source` directory, where `opensso-client-zip-root` is the directory where you unzipped `opensso-client-zip-root`.

Set your `JAVA_HOME` environment variable to JDK 1.5 or 1.4, depending on the version of the samples you are using.

- 1 On Solaris and Linux systems, make all shell scripts in the `opensso-client-zip-root/sdk/scripts` directory executable. For example:**

```
# cd opensso-client-zip-root/sdk/scripts
# chmod 755 *.sh
```

- 2 Compile the samples by executing the `scripts/compile-samples.sh` script.**

Note: You can invoke the sample scripts only from the `/sdk` parent directory and not directly from the `/scripts` directory.

- 3 Run the appropriate setup script for the samples: `scripts/setup.sh` on Solaris and Linux systems or `scripts/setup.bat` on Windows systems.**

Run the setup script only once for of the all Client SDK samples. The script will setup the `AMConfig.properties` file to point to the OpenSSO Enterprise server.

- 4 Run individual Client SDK samples by executing the shell or bat scripts in the `/scripts` directory. For example:**

```
# scripts/run-xacml-client-sample.sh
```

Note: At run time, a sample might require additional property files to be setup in the `/resources` directory. Check the comments included in each individual script for more information.

See Also For information about writing custom applications after you install the Client SDK, see Chapter 1, Enhancing Remote Applications Using the Client Software Development Kit, in the *Sun OpenSSO Enterprise 8.0 Developer's Guide*.

Configuring OpenSSO Enterprise Sessions

Sun™ OpenSSO Enterprise session configuration includes:

- [“Setting Session Quota Constraints” on page 91](#)
- [“Configuring Session Property Change Notifications” on page 94](#)

For other session attributes that you can configure, refer to the OpenSSO Enterprise Console online Help.

Setting Session Quota Constraints

The session quota constraints feature allows OpenSSO Enterprise to limit users to a specific number of active, concurrent sessions. An OpenSSO Enterprise administrator can set session quota constraints at the following levels:

- Globally. Constraints apply to all users.
- To an entity (organization or realm, role, or user). Constraints apply only to the specific users that belong to the entity.

Deployment Scenarios for Session Quota Constraints

The following OpenSSO Enterprise deployments support session quota constraints:

- OpenSSO Enterprise single server deployment
In this scenario, OpenSSO Enterprise is deployed on a single host server. OpenSSO Enterprise maintains the active session counts in memory for all logged in users. When a user attempts to log in to the server, OpenSSO Enterprise checks whether the number of the valid sessions for the user exceeds the session quota and then takes action based on the configured session quota constraints options.
- OpenSSO Enterprise session failover deployment

In this scenario, multiple instances of OpenSSO Enterprise are deployed on different host servers in a session failover configuration. The OpenSSO Enterprise instances are configured for session failover using Sun Java System Message Queue (Message Queue) as the communications broker and the Oracle Berkeley DB as the session store database. For more information about OpenSSO Enterprise session failover, see [Chapter 6, “Implementing OpenSSO Enterprise Session Failover.”](#)

In a session failover deployment, when a user attempts to log in, the OpenSSO Enterprise server receiving the session creation request first retrieves the session quota for the user from the OpenSSO Enterprise identity repository. Then, the OpenSSO Enterprise server fetches the session count for the user directly from the centralized session repository (accumulating all the sessions from all the OpenSSO Enterprise servers within the same site) and checks whether the session quota has been exhausted. If the session quota has been exhausted for the user, the OpenSSO Enterprise server takes action based on the configured session quota constraints options.

If session constraints are enabled in a session failover deployment and the session repository is not available, users (except superuser) are not allowed to log in.

In a session failover deployment, if an OpenSSO Enterprise instance is down, all the *valid* sessions previously hosted by that instance are still considered to be valid and are counted when the server determines the actual active session count for a given user. An OpenSSO Enterprise multiple server deployment that is not configured for session failover does not support session quota constraints.

Multiple Settings For Session Quotas

If a user has multiple settings for session quotas at different levels, OpenSSO Enterprise follows this precedence to determine the actual quota for the user:

- user (highest)
- role/organization/realm (based on the conflict resolution levels)
- global (lowest)

For example, Ken is a member of both the marketing and management roles. Session quotas are defined as follows (all have the same conflict resolution level):

- organization - 1
- marketing role - 2
- management role - 4
- user Ken - 3

Ken's quota is 3.

Configuring Session Quota Constraints

To configure session quota constraints, the top-level OpenSSO Enterprise administrator (such as amAdmin) must set specific attributes in the OpenSSO Enterprise Console for one of the OpenSSO Enterprise instances in your deployment.

▼ To Configure Session Quota Constraints

- 1 Log in to OpenSSO Enterprise Console as amAdmin.
- 2 Click Configuration, Global and then Session.
- 3 On the Session page, set Enable Quota Constraints to ON.
When this attribute is enabled, OpenSSO Enterprise enforces session quota constraints whenever a user attempts to log in as a new client and create a new session.
- 4 On the Session page, for each session attribute, either accept the default value or set a value as required for your deployment.
If you are configuring session property change notifications , see “[Configuring Session Property Change Notifications](#)” on page 94.

Read Timeout for Quota Constraint	<p>Specifies the time in milliseconds that an inquiry to the session repository for the active user session counts continues before timing out. If the maximum wait time is reached due to the unavailability of the session repository, the session creation request is rejected.</p> <p>Default: 6000 milliseconds</p>
Resulting Behavior If Session Quota Exhausted	<p>Determines the behavior if a user exhausts the session constraint quota. This attribute takes effect only if Enable Quota Constraints is enabled. Values can be:</p> <ul style="list-style-type: none">■ DENY_ACCESS. OpenSSO Enterprise rejects the login request for a new session.■ DESTROY_OLD_SESSION. OpenSSO Enterprise destroys the next expiring existing session for the same user and allows the new login request to succeed. <p>Default: DESTROY_OLD_SESSION</p>

Exempt Top-Level Admins From Constraint Checking	<p>Specifies whether session constraint quotas apply to the administrators who have the Top-level Admin Role. Takes effect only if the Enable Quota Constraints attribute is enabled.</p> <p>Default: NO</p> <p>The super user defined for OpenSSO Enterprise (<code>com.sun.identity.authentication.super.user</code>) is always exempt from session quota constraint checking.</p>
Deny User Login When Session Repository is Down	<p>Specifies whether a user can login if the session repository is down. Takes effect only if the Enable Quota Constraints attribute is enabled.</p> <p>Default: NO</p>
Maximum Session Time	<p>Specifies the time in minutes before a session expires and the user must re-authenticate to regain access. To balance the security requirements and convenience, consider setting the Max Session Time interval to a higher value and setting the Max Idle Time interval to a relatively low value.</p> <p>Default: 120 minutes</p>
Maximum Idle Time	<p>Specifies the idle time in minutes before a session expires and the user must re-authenticate to regain access.</p> <p>Default: 30 minutes</p>
Maximum Caching Time	<p>Specifies the time in minutes before a session contacts OpenSSO Enterprise to refresh cached session information. It is recommended that the Maximum Caching Time should always be less than the Maximum Idle Time.</p> <p>Default: 3 minutes</p>
Active User Sessions	<p>Specifies the maximum number of concurrent sessions for a user.</p> <p>Default: 5</p>

- 5 **When you have finished setting attributes, click Save.**
- If you reset any of these attributes, you must restart the server for the new values to take effect.

Configuring Session Property Change Notifications

The session property change notification feature causes OpenSSO Enterprise to send a notification to all registered listeners when a change occurs to a specific session property. This feature takes effect when **Enable Property Change Notifications** is enabled (ON) in the OpenSSO Enterprise Console.

For example, in a single sign-on (SSO) environment, one OpenSSO Enterprise session can be shared by multiple applications. When a change occurs on a specific session property defined in the “Notification Properties” list, OpenSSO Enterprise sends a notification to all registered listeners.

All client applications participating in the SSO automatically get the session notification if they are configured in the notification mode. The client cached sessions are automatically updated based on the new session state (including the change of any session property, if there is any).

An application that wants to take a specific action based on a session notification can write an implementation of the `SSOTokenListener` interface and then register the implementation through the `SSOToken.addSSOTokenListener` method. For more information, see the *Sun OpenSSO Enterprise 8.0 Developer's Guide*.

▼ To Configure Session Property Change Notifications

- 1 **Log in to the OpenSSO Enterprise Console as `amAdmin`.**
- 2 **Click `Configuration`, `Global` and then `Session`.**
- 3 **On the `Session` page, set `Enable Property Change Notifications` to `ON`.**
- 4 **On the `Session` page, add properties to the `Notification Properties` list.**

This list specifies the properties that cause OpenSSO Enterprise to send a notification to registered listeners when a change to a property occurs.

In `New Value`, add each property for which you want a notification sent when the property is changed, and then click `Add`.

- 5 **When you have finished adding properties to the list, click `Save`.**

Enabling the Access Manager (AMSDK) Identity Repository Plug-in

The Access Manager (AMSDK) Identity Repository (IdRepo) is a legacy plug-in that allows you to use the following OpenSSO Enterprise features with Sun Java System Directory Server (hence Directory Server):

- Role-based authentication
- Organization-based authentication
- Password reset service

This chapter describes how to enable the Access Manager Identity Repository plug-in, including:

- [“Requirements to Enable the Access Manager Identity Repository Plug-in” on page 97](#)
- [“Configuring Sun Java System Directory Server” on page 98](#)
- [“Configuring OpenSSO Enterprise Server” on page 99](#)
- [“Creating a Data Store Using the IdRepo Plug-in” on page 107](#)

Requirements to Enable the Access Manager Identity Repository Plug-in

The requirements to enable and use the Access Manager Identity Repository include:

- The `opensso.war` file must be deployed in a supported web container, and OpenSSO Enterprise server must be initially configured using either the GUI or command-line Configurator.
- Sun Java System Directory Server must be the OpenSSO Enterprise Identity Repository (user data store).

About the Examples: The examples in this chapter use the `ssoadm` command, which requires you to enter administrator passwords in password files. If you plan to use `ssoadm.jsp`, you must enter the passwords directly.

Configuring Sun Java System Directory Server

Configuring Directory Server involves loading the required object classes, attributes, and objects, which are available in the following LDIF files:

- *zip-root/opensso/ldif/sunone_schema2.ldif*
- *zip-root/opensso/ldif/ds_remote_schema.ldif*
- *config_dir/template/ldif/install.ldif*
- *zip-root/opensso/ldif/index.ldif*
- *zip-root/opensso/ldif/plugin.ldif*
- *zip-root/opensso/ldif/fam_sds_schema.ldif*

where:

- *zip-root* is where the *opensso.zip* file was unzipped.
- *config_dir* is the configuration directory specified during the initial configuration of *opensso.war*. For example: */opensso*

Configure Directory Server by loading the required object classes and attributes:

- [“To Configure an Existing Directory Server Identity Repository” on page 98](#)
- [“To Configure a New Directory Server Identity Repository” on page 99](#)

▼ To Configure an Existing Directory Server Identity Repository

This task describes how to configure an existing Directory Server Identity Repository that was previously deployed with Access Manager 7.1 or Access Manager 7 2005Q4, in either legacy or realm mode.

1 In the *fam_sds_schema.ldif* file, replace the following items:

- *@NORMALIZED_RS@* with the normalized root suffix. For example: *o=example,o=isp*
- *@RS_RDN@* with the relative DN of the root suffix. For example: *example*
- *@ORG_NAMING_ATTR@* with the organization naming attribute. For example: *o*
- *@ADMIN_PWD@* with the passwords for *dsameuser* and *puser*.
- *@AMLDAPUSERPASSWD@* with the password for *amldapuser*.

2 Load the following object classes to the Directory Server schema from the *fam_sds_schema.ldif* file:

- *sunFederationManagerDataStore*
- *sunFMSAML2NameIdentifier*

To load these object classes, use the Directory Server Console, Directory Service Command Center (DSCC), or a command-line utility such as `ldapmodify`.

- 3 Continue with [“Configuring OpenSSO Enterprise Server” on page 99](#).

▼ To Configure a New Directory Server Identity Repository

This task describes how to configure a new Directory Server Identity Repository.

- 1 In the LDIF files, replace the following items:

- `@NORMALIZED_RS@` with the normalized root suffix. For example: `o=example,o=isp`
- `@RS_RDN@` with the relative DN of the root suffix. For example: `example`
- `@ORG_NAMING_ATTR@` with the organization naming attribute. For example: `o`
- `@ADMIN_PWD@` with the passwords for `dsameuser` and `puser`.
- `@AMLDAUSERPASSWD@` with the password for `amldapuser`.

- 2 Load the following LDIF files, in the order shown:

- `zip-root/opensso/ldif/sunone_schema2.ldif`
- `zip-root/opensso/ldif/ds_remote_schema.ldif`
- `config_dir/template/ldif/install.ldif`
- `zip-root/opensso/ldif/index.ldif`
- `zip-root/opensso/ldif/plugin.ldif`
- `zip-root/opensso/ldif/fam_sds_schema.ldif`

To load these LDIF files, use the Directory Server Console, Directory Service Command Center (DSCC), or a command-line utility such as `ldapmodify`.

Configuring OpenSSO Enterprise Server

OpenSSO Enterprise server must be configured for the Access Manager Identity Repository plug-in. The steps you follow depend on the values of your user and organization naming attributes:

- **Scenario 1:** Your user naming attribute is `uid` **and** your organization naming attribute is `o`. Continue with [“To Configure OpenSSO Enterprise Server Using the `ssoadm` Command” on page 100](#).
- **Scenario 2:** Your user naming attribute is **not** `uid` **or** your organization naming attribute is **not** `o`. Continue with [“Configure OpenSSO Enterprise Server Manually” on page 100](#).

▼ To Configure OpenSSO Enterprise Server Using the `ssoadm` Command

In this scenario, the `ssoadm` command with the `add-amsdk-idrepo-plugin` subcommand configures OpenSSO Enterprise server to enable the Access Manager Identity Repository by performing all of these tasks:

- Loads the Directory Access Instructions (DAI) service
- Adds the IdRepo subschema (`sunIdentityRepositoryService`)
- Updates the Directory Server information in `serverconfig.xml`
- Enables persistent searches for the Access Manager SDK (AMSDK)

1 Execute the `ssoadm` command with the `add-amsdk-idrepo-plugin` subcommand. For example:

```
# ./ssoadm add-amsdk-idrepo-plugin -u amadmin -f ./password-file
-b "dc=example,dc=com" -s ldaphost.example.com:389
-x ./dsamepassword -p ./proxypassword
```

where:

- u specifies the administrative user. For example: `amadmin`
- f specifies the password file for the administrative user.
- b specifies the base dn. For example: `dc=example,dc=com`
- s specifies the directory server host, port, and protocol. Examples for the `-s` option are:
 - `ldap://host:port`
 - `host:port` (The protocol defaults to `ldap`.)
 - `host` (The protocol defaults to `ldap`, and the port defaults to 389.)
- x specifies the password file for `dsameuser`.
- p specifies the password file for `proxypuser`.

2 Restart the OpenSSO Enterprise server web container.

3 Continue with [“Creating a Data Store Using the IdRepo Plug-in” on page 107](#).

Configure OpenSSO Enterprise Server Manually

In this scenario, you must configure OpenSSO Enterprise server manually by:

- [“Loading the Directory Access Instructions \(DAI\) Service” on page 101](#)
- [“Loading the Access Manager SDK \(AMSDK\) Subschema” on page 101](#)
- [“Updating the Directory Server Information for the AMSDK Plug-in” on page 105](#)
- [“Enabling Persistent Search Connections for the IdRepo Plug-in” on page 106](#)

Loading the Directory Access Instructions (DAI) Service

▼ To Load the DAI Service

- 1 In the `zip-root/opensso/xml/ums.xml` file, replace the following items, as needed for your deployment:
 - @USER_NAMING_ATTR@ with your user naming attribute
 - @ORG_NAMING_ATTR@ with your organization naming attribute
- 2 Load the DAI service from the `ums.xml` file using the `ssoadm` command with the `create-svc` subcommand. For example:

```
# ./ssoadm create-svc -u amadmin -f ./password-file
--xmlfile zip-root/opensso/xml/ums.xml
```

where:

-u specifies the administrative user. For example: amadmin

-f specifies the password file for the administrative user.

--xmlfile (or -X) specifies the path to the `ums.xml` file.

`zip-root` is where the `opensso.zip` file was unzipped.

Loading the Access Manager SDK (AMSDK) Subschema

▼ To Load the AMSDK Subschema

- 1 In `zip-root/opensso/xml/idRepoAmSDK.xml`, replace @NORMALIZED_ORGBASE@ with the Directory Server root suffix.
- 2 Load the `IdRepo` subschema using the `ssoadm` command with the `add-sub-schema` subcommand. For example:

```
# ./ssoadm add-sub-schema -u amadmin -f ./password-file \
-s sunIdentityRepositoryService -t Organization -F idRepoAmSDK.xml
```

where:

-u specifies the administrative user. For example: amadmin

-f specifies the password file for the administrative user.

-s specifies the service name. Must be `sunIdentityRepositoryService`

-t specifies the schema type. Must be: `Organization`

-F specifies the path to the `idRepoAmSDK.xml` file that you created earlier.

IdRepo Plug-in SubSchema

Use the following entries to create the `idRepoAmSDK.xml` file.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!--
DO NOT ALTER OR REMOVE COPYRIGHT NOTICES OR THIS HEADER.

Copyright (c) 2006 Sun Microsystems Inc. All Rights Reserved

The contents of this file are subject to the terms
of the Common Development and Distribution License
(the License). You may not use this file except in
compliance with the License.

You can obtain a copy of the License at
https://opensso.dev.java.net/public/CDDLv1.0.html or
opensso/legal/CDDLv1.0.txt
See the License for the specific language governing
permission and limitations under the License.

When distributing Covered Code, include this CDDL
Header Notice in each file and include the License file
at opensso/legal/CDDLv1.0.txt.
If applicable, add the following below the CDDL Header,
with the fields enclosed by brackets [] replaced by
your own identifying information:
"Portions Copyright [year] [name of copyright owner]"

$Id$
-->

<!DOCTYPE ServicesConfiguration
PUBLIC "-//iplanet//Service Management Services (SMS) 1.0 DTD//EN"
"jar://com/sun/identity/sm/sms.dtd">

<ServicesConfiguration>
  <Service name="sunIdentityRepositoryService" version="1.0">
    <Schema>
      <Organization>
        <!-- Schema for linking SMS's organization to
        AMSDK's organization. -->
        <SubSchema name= "amSDK"
          inheritance = "multiple"
          i18nKey = "a101">
```

```

<!-- The two attributes below, if defined, allow the IdRepo
      SDK to map attribute names and to determine entities
      supported and operations supported by this plugin. If
      no mapping is provided, the assumption is that the
      attribute names are the same (no mapping needed. If the
      list of entities supported is not provided then the
      assumption is that no entities are supported -->

<AttributeSchema name="sunIdRepoClass"
  type = "single"
  syntax = "string"
  any="required"
  validator="RequiredValueValidator"
  i18nKey = "a102">
  <DefaultValues>
    <Value>com.ipplanet.am.sdk.AMSDKRepo</Value>
  </DefaultValues>
</AttributeSchema>
<AttributeSchema name="RequiredValueValidator"
  type="validator"
  syntax="string">
  <DefaultValues>
    <Value>com.sun.identity.sm.RequiredValueValidator</Value>
  </DefaultValues>
</AttributeSchema>
<AttributeSchema name="sunIdRepoNamingAttribute"
  type="list"
  syntax="string"
  i18nKey="">
  <DefaultValues>
    <Value>user=uid</Value>
    <Value>group=cn</Value>
    <Value>role=cn</Value>
    <Value>agent=uid</Value>
    <Value>filteredrole=cn</Value>
  </DefaultValues>
</AttributeSchema>
<AttributeSchema name="sunIdRepoAttributeMapping"
  type="list"
  syntax="string"
  i18nKey="">
</AttributeSchema>
<AttributeSchema name="sunIdRepoSupportedOperations"
  type="list"
  syntax="string"
  validator="validatatorClass for verifying this list against supported list"
  i18nKey="">

```

```
<DefaultValues>

  <!-- IdRepos can provide the object types
  it can manage and the operations that can
  be performed on these objects, as a key-value
  pair. Object types could be User, Group, Role,
  etc., and the operations are limited to create,
  read, edit and modify. The values are case
  insensitive. -->

  <Value>user=read,create,edit,delete,service</Value>
  <Value>agent=read,edit</Value>
  <Value>role=read,edit,service</Value>
  <Value>filteredrole=read,edit,service</Value>
  <Value>group=read</Value>
</DefaultValues>
</AttributeSchema>

<!-- From here on the list of plugin specific attributes
for determining things like organization relating to
this realm, authentication source,
user containers, group containers etc etc. -->

<AttributeSchema name="amSDKOrgName"
  type = "single"
  syntax = "string"
  i18nKey = "a106">
  <DefaultValues>
    <Value>@NORMALIZED_ORGBASE@</Value>
  </DefaultValues>
</AttributeSchema>
<AttributeSchema name="sun-idrepo-amSDK-config-people-container-name"
  type="single"
  syntax="string"
  i18nKey="a2053">
  <DefaultValues>
    <Value>ou</Value>
  </DefaultValues>
</AttributeSchema>

<AttributeSchema name="sun-idrepo-amSDK-config-people-container-value"
  type="single"
  syntax="string"
  i18nKey="a2054">
  <DefaultValues>
    <Value>people</Value>
  </DefaultValues>
</AttributeSchema>
```



```

<AttributeSchema name="sun-idrepo-amSDK-config-agent-container-name"
  type="single"
  syntax="string"
  i18nKey="a2055">
  <DefaultValues>
    <Value>ou</Value>
  </DefaultValues>
</AttributeSchema>

<AttributeSchema name="sun-idrepo-amSDK-config-agent-container-value"
  type="single"
  syntax="string"
  i18nKey="a2056">
  <DefaultValues>
    <Value>agents</Value>
  </DefaultValues>
</AttributeSchema>

<AttributeSchema name="sun-idrepo-amSDK-config-recursive-enabled"
  type="single"
  syntax="boolean"
  i18nKey="a2057">
  <DefaultValues>
    <Value>>false</Value>
  </DefaultValues>
</AttributeSchema>

<AttributeSchema name="sun-idrepo-amSDK-config-copyconfig-enabled"
  type="single"
  syntax="boolean"
  i18nKey="a2058">
  <DefaultValues>
    <Value>>false</Value>
  </DefaultValues>
</AttributeSchema>
</SubSchema>
</Organization>
</Schema>
</Service>
</ServicesConfiguration>

```

Updating the Directory Server Information for the AMSDK Plug-in

Update the Directory Server information by exporting, modifying, and then re-importing the information.

Important: If your deployment has multiple OpenSSO Enterprise server instances, you must perform the following steps on all server instances.

▼ To Update the Directory Server Information for the AMSDK Plug-in

- 1 **Export the Directory Server configuration information from the OpenSSO Enterprise server instance using the `ssoadm` command with the `get-svccfg-xml` subcommand. For example:**

```
# ./ssoadm get-svrcfg-xml -u amadmin -f ./password-file \  
-s http(s)://host.domain:port/opensso -o serverconfig.xml
```

where:

- u specifies the administrative user. For example: amadmin
- f specifies the password file for the administrative user.
- s specifies the server instance name. For example:
`https://openssohost1.example.com:8080/opensso`
- o specifies the output file name that will contain the Directory Server configuration information. For example: `serverconfig.xml`

- 2 **Edit the Directory Server configuration information in the `serverconfig.xml` file as follows:**

- a. **In the `<ServerGroup name="default" ...>` entry, add the Directory Server configuration information, including the host, port and protocol.**
- b. **Update the encrypted passwords for the admin and proxy users. Use the `ampassword` utility to obtain the encrypted passwords**

- 3 **Import the revised Directory Server configuration information using the `ssoadm` command with the `set-svccfg-xml` subcommand. For example:**

```
# ./ssoadm set-svrcfg-xml -u amadmin -f ./password-file  
-s http(s)://host.domain:port/opensso -X serverconfig.xml
```

where:

- u specifies the administrative user. For example: amadmin
- f specifies the password file for the administrative user.
- s specifies the server instance name. For example:
`http://openssohost1.example.com:8080/opensso`
- X specifies the input file name that contains the revised Directory Server configuration information. For example: `serverconfig.xml`

Enabling Persistent Search Connections for the IdRepo Plug-in

This task involves enabling the persistent search (`psearch`) connections for the OpenSSO Enterprise server to allow the Access Manager Identity Repository to receive change notifications.

▼ To Enable Persistent Search Connections for the Access Manager Identity Repository

- 1 Log in to the OpenSSO Enterprise Admin Console.
- 2 Click **Configuration** and then **Servers and Sites**.
- 3 For each OpenSSO server instance listed:
 - a. Click **SDK** and then **Event Service**.
 - b. Remove the entries in **Disabled Event Service Connection**. This field must be empty.
 - c. Click **Save**.
- 4 Log out of the Console.
- 5 Restart the OpenSSO Enterprise server web container.

Creating a Data Store Using the IdRepo Plug-in

Use the following procedure to create a new data store or to verify that you correctly enabled the Access Manager Identity Repository.

▼ To Create a Data Store Using the IdRepo Plug-in

- 1 Log in to the OpenSSO Enterprise Admin Console.
- 2 Click **Access Control** and then the realm.
- 3 Click **Data Stores**.
- 4 Click **New**, and you should see the option for the **Access Manager Identity Repository Plug-in**.
- 5 Enter the **Data Store Name**, and click **Next** to continue the configuration.
(Or, if you are not actually creating a new data store, click **Cancel**.)
- 6 If you are creating a new data store, continue with the configuration and click **Finish** when you are done.

Managing LDAP Persistent Searches

OpenSSO Enterprise uses LDAP persistent searches (psearches) to obtain asynchronous notifications of changes that occur in Sun Java System Directory Server.

- [“Creation of Persistent Searches” on page 109](#)
- [“Disabling Persistent Searches” on page 110](#)
- [“Configuration Properties That Affect Persistent Searches” on page 113](#)

Creation of Persistent Searches

The OpenSSO Enterprise Event Service (`amEventService`) creates and manages the persistent search connections. By default, the Event Service creates the following persistent search connections during the OpenSSO Enterprise server startup:

- `aci` - To receive changes to the `aci` attribute, with the persistent search using the LDAP filter (`aci=*`).
- `sm` - To receive changes in the OpenSSO Enterprise configuration data store (service management node), which includes objects with the `sunService` or `sunServiceComponent` marker object class. For example, creation of a new policy to define access privileges for a protected resource or changes to the rules, subjects, conditions, or response providers for an existing policy.

- `um` - To receive changes in the user data store (user management node). For example, changes to a user's name or address.

The Directory Server `nsslapd-maxpsearch` attribute defines the maximum number of persistent searches that can be performed on Directory Server. For example:

Property Value Entry DN: `cn=config`
Valid Range: 1 to maximum threadnumber
Default Value: 30
Syntax: Integer
Example: `nsslapd-maxpsearch: 30`

The Directory Server `nsIdletimeout` attribute does not apply to these connections. A connection is closed when the Directory Server or OpenSSO Enterprise server goes down or after a load balancer or firewall TCP timeout.

Disabling Persistent Searches

Each active persistent search requires an open TCP connection between OpenSSO Enterprise server and Directory Server, which can cause a performance overhead on Directory Server. Therefore, use persistent searches only for essential tasks and close any idle LDAP connections when they are no longer required.

If you determine that improving performance is critical for your deployment, the `com.sun.am.event.connection.disable.list` property allows you to disable persistent searches.



Caution – Before disabling a persistent search, however, you should understand the consequences. The `com.sun.am.event.connection.disable.list` property was introduced primarily to avoid overhead on Directory Server when multiple version 2.1 J2EE policy agents were used, because each agent established persistent searches. OpenSSO Enterprise does not support version 2.1 policy agents, and version 2.2 and version 3.0 J2EE policy agents do not establish persistent searches.

A component with a disabled persistent search does not receive notifications from Directory Server. Consequently, changes made in Directory Server are not be notified to the component cache, and the component cache can go stale. For example, if you disable persistent searches for changes in the user data store (um), OpenSSO Enterprise server does not receive notifications from Directory Server for any changes to the user data store. Therefore, an agent does not get notifications from OpenSSO Enterprise to update its local user cache with any new values for user attributes. Then, if an application queries the agent for user attributes, the application might receive old values for the attributes.

Disabling persistent searches for a component is recommended only if absolutely required for a deployment. For example, if you know that changes to the configuration data store (service management (sm) node) will not happen in an environment, you can disable the persistent search for this component. However, if any changes do occur for any of the services, a server restart is required to get the changes. This situation also applies to persistent searches for changes to the aci attribute and user data store (sm).

▼ To Disable Persistent Searches Using the Console

- 1 Log in to the Admin Console as `amadmin`.
- 2 Click Configuration, Servers and Sites, *server-name*, SDK, and then Event Service.
- 3 In the Disable Event Service Connection field, specify `aci`, `sm`, or `um` (or a combination, with each item separated by a comma).
- 4 Click Save and log out of the Console.
- 5 Restart the OpenSSO Enterprise web container.

Disabling Persistent Searches by Setting the `com.sun.am.event.connection.disable.list` Property

You can also disable persistent searches by setting the `com.sun.am.event.connection.disable.list` property, using the `ssoadm` command, to one or more of the following values: `aci`, `sm`, or `um`.

Values are case insensitive. To specify multiple values, separate each value with a comma. For example:

To disable all persistent search connections:

```
com.sun.am.event.connection.disable.list=aci,sm,um
```

To disable persistent searches for ACI changes only:

```
com.sun.am.event.connection.disable.list=aci
```

To disable persistent searches for configuration data store changes only:

```
com.sun.am.event.connection.disable.list=sm
```

To disable persistent searches for user data store changes only:

```
com.sun.am.event.connection.disable.list=um
```

To disable persistent searches for configuration data store and user data store changes:

```
com.sun.am.event.connection.disable.list=sm,um
```

Re-Enabling Persistent Searches

If you need to re-enable a persistent search that you have disabled, follow the instructions in the previous section using the Admin Console, however, leave a blank for the search (or searches) you want to re-enable.

You can also re-enable one or more persistent searches by setting the `com.sun.am.event.connection.disable.list` property to a blank value for each specific search you want to re-enable. For example, to re-enable the search for configuration data store and `aci` changes, but leave the search disabled for user data store changes, set the property as follows:

```
com.sun.am.event.connection.disable.list=um
```

When you are finished, restart the OpenSSO Enterprise web container

Configuration Properties That Affect Persistent Searches

Set these properties either in the OpenSSO Enterprise Admin Console or using the `ssoadm` command.

Connection Idle Timeout

- `com.sun.am.event.connection.idle.timeout` specifies the number of minutes after which persistent searches will be restarted. The default is 0, which indicates that persistent searches will not be restarted.

If persistent search connections are made through a load balancer or firewall, these connections are subject to the TCP time out value of the load balancer or firewall. If the load balancer or firewall closes the persistent search connection due to an idle TCP time out, change notifications are not sent to OpenSSO Enterprise unless the persistent search connection is re-established.

Therefore, set `com.sun.am.event.connection.idle.timeout` to a value lower than the load balancer or firewall TCP timeout, to make sure that persistent searches are restarted before the connections are dropped. The difference between the load balancer or firewall timeout value should not be more than 5 minutes. For example, if your load balancer idle connection time out is 50 minutes, set `com.sun.am.event.connection.idle.timeout` to 45 minutes.

Persistent Search Connection Restart

- `com.iplanet.am.event.connection.num.retries` specifies the number of attempts to successfully re-establish the persistent search connections. The default is 3.
- `com.iplanet.am.event.connection.delay.between.retries` specifies the delay in milliseconds between retries to re-establish the persistent search connections. The default is 3000.
- `com.iplanet.am.event.connection.ldap.error.codes.retries` specifies the LDAP exception error codes for which retries to re-establish persistent search connections will trigger. The default error codes are 80,81,91; however, you can specify any valid LDAP error code.

These four properties apply only to the persistent search (Event Service) connections and are not shared by other modules. For example, these properties do not affect the SDK LDAP connection pool or the authentication LDAP or policy LDAP connections.

LDAPv3 Plug-in Idle Timeout

Each instance of an LDAPv3 plug-in data store creates a persistent search connection using the filter (`objectclass=*`). Therefore, exercise caution in creating LDAPv3 data stores to prevent the OpenSSO Enterprise server from being flooded with too many notifications. Also, Directory Server does not return an error if the base DN of the persistent search does not exist, so make sure you supply the correct base DN.

- `sun-idrepo-ldapv3-config-idletimeout` specifies the maximum idle time before an LDAPv3 data store restarts a persistent search connection. If you are using a load balancer or firewall, set this value lower than the load balancer or firewall TCP connection idle timeout value.

For information about using persistent searches in custom applications, see the *OpenSSO Enterprise 8.0 Developers Guide*.

Customizing OpenSSO Enterprise Administration Console Pages

For some deployments, you might need to customize Administration Console pages such as the login or logout page. For example, you might want to add a warning banner to the login page.

Customizing the OpenSSO Enterprise Login and Logout Pages

▼ To Customize the OpenSSO Enterprise Login and Logout Pages

- 1 Make sure that your `JAVA_HOME` environment variable points to JDK 1.5 or later.
- 2 Download and unzip `opensso.zip`, as described in [“Downloading OpenSSO Enterprise” on page 23](#).
- 3 Create a new staging directory and extract the files from `opensso.war` in this staging directory. For example:

```
# mkdir customwar
# cd customwar
# jar xvf zip-root/opensso/deployable-war/opensso.war
```

where `zip-root` is where you unzipped `opensso.zip`.
- 4 Customize the pages as required for your deployment by editing the JSP and XML files required by your deployment. For example:

`customwar/config/auth/default/Login.jsp`

Console login page

<code>customwar/config/auth/default/Logout.jsp</code>	Console logout page
<code>customwar/config/auth/default/DataStore.xml</code>	Data store authentication page
<code>customwar/config/auth/default/LDAP.xml</code>	LDAP authentication page

where `customwar` is where you extracted the files from `opensso.war`.

5 Generate a new WAR file with the customized files. For example:

```
# cd customwar
# jar uvf OpenSSOCustom.war *
```

6 Deploy the new customized war file, as described in [“Deploying the OpenSSO Enterprise WAR File” on page 28](#).

Loading the OpenSSO Schema into Sun Java System Directory Server

In this scenario, you want to use Sun Java System Directory Server as the OpenSSO Enterprise user data store, but the required OpenSSO schema files have not been loaded into Directory Server.

If you configuring OpenSSO Enterprise using the Configurator, you can simply specify Directory Server as the user data store, as described in [Chapter 3, “Configuring OpenSSO Enterprise Using the GUI Configurator,”](#) or [Chapter 4, “Configuring OpenSSO Enterprise Using the Command-Line Configurator.”](#)

Otherwise, you must load the OpenSSO schema and index LDIF files using the Directory Server Console, Directory Service Command Center (DSCC), or a command-line utility such as `ldapmodify`, depending on the version of Directory Server you are using and your personal preference.

The LDIF files you must load into Directory Server are:

- `zip-root/opensso/ldif/sunone_schema2.ldif`
- `zip-root/opensso/ldif/ds_remote_schema.ldif`
- `zip-root/opensso/ldif/fam_sds_schema.ldif`
- `zip-root/opensso/ldif/fam_sds_index.ldif`
- `zip-root/opensso/ldif/index.ldif`
- `zip-root/opensso/ldif/plugin.ldif`

where *zip-root* is the directory where you unzipped `opensso.zip`.

Loading the OpenSSO Schema into Directory Server

▼ To Load the OpenSSO Schema into Directory Server

- 1 In `fam_sds_index.ldif` and `index.ldif`, replace `@DB_NAME@` with the name of your backend DB.

The index creation LDIF files require the backend DB name. Both `index.ldif` and `fam_sds_index.ldif` contain the `@DB_NAME@` variable to represent the backend DB name of the specific instance where you originally deployed OpenSSO Enterprise.

For example, a system with the `dc=openssohost,dc=example,dc=com` suffix, an index entry might look like:

```
dn: cn=nsroledn,cn=index,cn=openssohost,cn=ldbm database,cn=plugins,cn=config
dn: cn=memberof,cn=index,cn=openssohost,cn=ldbm database,cn=plugins,cn=config
dn: cn=iplanet-am-static-group-dn,cn=index,cn=openssohost,cn=ldbm database,cn=plugins,cn=config
dn: cn=iplanet-am-modifiable-by,cn=index,cn=openssohost,cn=ldbm database,cn=plugins,cn=config
dn: cn=sunxmlkeyvalue,cn=index,cn=openssohost,cn=ldbm database,cn=plugins,cn=config
dn: cn=o,cn=index,cn=openssohost,cn=ldbm database,cn=plugins,cn=config
dn: cn=ou,cn=index,cn=openssohost,cn=ldbm database,cn=plugins,cn=config
dn: cn=sunPreferredDomain,cn=index,cn=openssohost,cn=ldbm database,cn=plugins,cn=config
dn: cn=associatedDomain,cn=index,cn=openssohost,cn=ldbm database,cn=plugins,cn=config
dn: cn=sunOrganizationAlias,cn=index,cn=openssohost,cn=ldbm database,cn=plugins,cn=config
```

- 2 Load the LDIF files into Directory Server, in this order:

- `sunone_schema2.ldif`
- `ds_remote_schema.ldif`
- `fam_sds_schema.ldif`
- `fam_sds_index.ldif`
- `index.ldif`
- `plugin.ldif`

For example, using `ldapmodify`:

```
ldapmodify -h dshost -p dsport -D "cn=directory manager" -w dmpasswd -c -f sunone_schema2.ldif
ldapmodify -h dshost -p dsport -D "cn=directory manager" -w dmpasswd -c -f ds_remote_schema.ldif
ldapmodify -h dshost -p dsport -D "cn=directory manager" -w dmpasswd -c -f fam_sds_schema.ldif
ldapmodify -h dshost -p dsport -D "cn=directory manager" -w dmpasswd -c -a -f fam_sds_index.ldif
ldapmodify -h dshost -p dsport -D "cn=directory manager" -w dmpasswd -c -a -f index.ldif
ldapmodify -h dshost -p dsport -D "cn=directory manager" -w dmpasswd -c -a -f plugin.ldif
```

where `dshost` and `dsport` are the Directory Server host name and port, and `dmpasswd` is the Directory Manager password.

Note: If you encounter a SASL BIND error use the `-x` option with `ldapmodify`.

- 3 Create the base container entries to prepare the DIT for the default use data store configuration as created by OpenSSO Enterprise server:
 - a. Create the base container entries in the `/net/slaped.dshost.domain/export/share/ldif/ldapentries` file by replacing `dc=sun,dc=com` with your root suffix.
 - b. Execute the following command:


```
ldapmodify -h dshost -p dsport -D "cn=directory manager" -w dmpasswd -c -a -f
          ldapentries
```
- 4 In the OpenSSO Enterprise Admin Console, create a new data store with the Directory Server that you just configured with the OpenSSO schema:
 - a. Log in to the Console as `amadmin`.
 - b. Click Access Control, *realm-name*, Data Stores, and then New.
 - c. Specify the new data store Name.
 - d. Specify the Type as Sun DS with OpenSSO schema and then click Next.
 - e. Specify the LDAP Bind DN as `cn=dsameuser,ou=dsame users,root-suffix`, where *root-suffix* is the root suffix for the Directory Server user data store.
 - f. Specify other values as required for your deployment, and then click Finish.

Next Steps You can now use Directory Server as the OpenSSO Enterprise 8.0 user data store.

Uninstalling OpenSSO Enterprise

- “Uninstalling OpenSSO Enterprise Server” on page 121
- “Uninstalling the OpenSSO Enterprise Utilities and Scripts” on page 122
- “Uninstalling a Distributed Authentication UI Server Deployment” on page 122
- “Uninstalling an IDP Discovery Deployment” on page 123
- “Uninstalling a Client Sample Deployment” on page 123
- “Uninstalling a Fedlet Deployment” on page 124
- “Uninstalling an OpenSSO Enterprise Console Only Deployment” on page 124
- “Uninstalling the OpenSSO Enterprise Client SDK” on page 125
- “Removing OpenSSO Enterprise Entries From Directory Server” on page 125

Uninstalling OpenSSO Enterprise Server

This scenario applies to a full OpenSSO Enterprise server deployment and an OpenSSO Enterprise server only (no console) deployment.

▼ To Uninstall OpenSSO Enterprise Server

- 1 Undeploy `opensso.war` in the web container using the web container administration console or command-line utility.
- 2 On Windows systems, you might first need to stop the web container to perform the following steps.
- 3 Remove the following directories and all of their contents:
 - *ConfigurationDirectory* is the directory created when the OpenSSO Enterprise instance is initially configured using the Configurator.

The default is `opensso` in the home directory of the user running the Configurator. If the Configurator is run by root, *ConfigurationDirectory* is `/opensso`.

- `user-home-directory.openssocfg` where *user-home-directory* is the home directory of the user who deployed the `opensso.war` file. If this user is root, the directory is `/ .openssocfg`.
- 4 **Optionally, remove the `opensso.zip` and extracted files.**
- 5 **If the OpenSSO Enterprise server instance was using the OpenSSO data store, the data store port is in use by the LISTEN socket. If you need to release this port, follow these steps:**
 - a. **Stop the web container server instance or domain to release the port.**
 - b. **Check the data store port. For example, if the default port 50389 is used:**

```
netstat -a | grep 50389
```

The port should not be in use for the LISTEN socket.
 - c. **Restart the web container server instance or domain.**

Uninstalling the OpenSSO Enterprise Utilities and Scripts

▼ To Uninstall the OpenSSO Enterprise Utilities and Scripts

- 1 **Remove the directory and its contents where `openssoAdminTools.zip` was extracted.**
- 2 **Optionally, remove the `openssoAdminTools.zip` file.**

Uninstalling a Distributed Authentication UI Server Deployment

▼ To Uninstall a Distributed Authentication UI Server Deployment

- 1 **Undeploy the Distributed Authentication UI server WAR file in the web container using the web container administration console or command-line utility.**
- 2 **On Windows systems, you might first need to stop the web container to perform the following steps.**

- 3 Remove the /FAMDistAuth directory including the AMDistAuthConfig.properties configuration file.**

The /FAMDistAuth directory is located in the home directory of the user running the web container on which the Distributed Authentication UI WAR file is deployed.

- 4 Remove the debug directory and its contents.**

The location of the debug directory was specified when the Distributed Authentication UI server was configured using the Configurator.

Uninstalling an IDP Discovery Deployment

▼ To Uninstall an IDP Discovery Deployment

- 1 Undeploy the IDP Discovery WAR in the web container.**
- 2 On Windows systems, you might first need to stop the web container to perform the following steps.**
- 3 Remove the libIDPDiscoveryConfig.properties file under the home directory of the user running the web container.**
- 4 Remove the debug directory and its contents.**

The location of the debug directory was specified when the IDP Discovery deployment was configured using the Configurator..

Uninstalling a Client Sample Deployment

▼ To Uninstall a Client Sample Deployment

- 1 Undeploy the client sample WAR in the web container.**
- 2 On Windows systems, you might first need to stop the web container to perform the following steps.**
- 3 Remove the AMConfig.properties file under the home directory of the user running the web container.**

4 Remove the debug directory and its contents.

The location of the debug directory was specified when the client sample was configured.

5 Remove these files:

- `ClientSampleWSC.properties`
- Discovery resource offering files, which begin with `RO_` and are located under the home directory of the user running the web container.

Uninstalling a Fedlet Deployment

▼ To Uninstall a Fedlet Deployment

1 Undeploy the `fedlet.war` in the web container.

2 On Windows systems, you might first need to stop the web container to perform the next step.

3 Remove the `fedlet` configuration directory.

By default, the `fedlet` directory is located under the user's home directory.

Uninstalling an OpenSSO Enterprise Console Only Deployment

▼ To Uninstall an OpenSSO Enterprise Console Only Deployment

1 Undeploy `opensso.war` in the web container using the web container administration console or command-line utility.

2 On Windows systems, you might first need to stop the web container to perform the following steps.

3 Remove the `AMConfig.properties` file under home directory of the user running the web container.

4 Remove the debug directory.

The location of the debug directory was specified when the console only deployment was configured using the Configurator.

Uninstalling the OpenSSO Enterprise Client SDK

▼ To Uninstall the OpenSSO Enterprise Client SDK

1 Remove the directory where the `opensso-client.zip` file was extracted.**2 Remove the client SDK debug directory.**

The client SDK debug directory was specified when one of the following setup scripts was run:

- Solaris and Linux systems: `scripts/setup.sh`
- Windows systems: `scripts/setup.bat`

3 Optionally, remove the `opensso-client.zip` file.

Removing OpenSSO Enterprise Entries From Directory Server

If you used Sun Java System Directory Server as either the configuration data store or user data store, you must manually remove the OpenSSO Enterprise entries.

To remove these entries, use the Directory Server Console, Directory Service Command Center (DSCC), or a command-line utility such as `ldapmodify`.

▼ To Remove OpenSSO Enterprise Entries From Directory Server

1 Remove the OpenSSO Enterprise schema and attribute index entries, which are loaded during the OpenSSO Enterprise installation from the following files:

- `am_sm_ds_schema.ldif`
- `ds_remote_slds_schema.ldif`
- `index.ldif`
- `fam_sds_schema.ldif`
- `fam_sds_index.ldif`

2 If Directory Server is the configuration data store, remove the entire `ou=services` sub-branch, which is under the root suffix.

3 Depending on the features you used, remove OpenSSO Enterprise user entries from the user data store.

For example, federation attributes (`sun-fm-saml2-nameid-infokey` and `sun-fm-saml2-nameid-info`) might be added to the user entries if you used SAMLv2 single sign-on (SSO). To determine which entries you need to remove, search the user entries for the schema attributes found in these LDIF files.

- `ds_remote_slds_schema.ldif`
- `fam_sds_schema.ldif`

Index

A

- amsfo script, 61
- amsfopassword script, 60
- audience for this guide, 9

D

- documentation
 - Access Manager, 10-11
 - collections, 12
 - related product, 12

G

- guest user, Message Queue, 59

I

- imqusermgr command, Message Queue, 59

O

- organization of this guide, 10

P

- prerequisites for this guide, 9-10
- publish/subscribe, Message Queue, 56

R

- related guides, 10-12

S

- session property change notification, 94
- session quota constraints, 91

