Sun OpenSSO Enterprise 8.0 Installation and Configuration Guide

Part No: 820–3320–16
August 10, 2010

100811@24661

Contents

Preface

The *Sun OpenSSO Enterprise 8.0 Installation and Configuration Guide* describes how to install and configure OpenSSO Enterprise 8.0, including OpenSSO Enterprise, OpenSSO Enterprise 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

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, Oracle WebLogic Server, 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 by chapters, as described in the Contents.

Related Documentation

Related documentation is available as follows:

- "OpenSSO Enterprise Documentation Set" on page 12
- "Policy Agent Documentation" on page 13
- "Related Product Documentation" on page 13

OpenSSO Enterprise Documentation Set

The following table describes the OpenSSO Enterprise documentation set, which is available on the following collection: `http://docs.sun.com/coll/1767.1`

TABLE P–1 OpenSSO Enterprise Documentation Set

Title	Description
Sun OpenSSO Enterprise 8.0 Release Notes	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.
Sun OpenSSO Enterprise 8.0 installation and Configuration Guide (this guide)	Provides information about installing and configuring OpenSSO Enterprise.about, including OpenSSO Enterprise server, Administration Console only, client SDK, scripts and utilities, Distributed Authentication UI server, and session failover.
Sun OpenSSO Enterprise 8.0 Technical Overview	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.
Sun OpenSSO Enterprise 8.0 Deployment Planning Guide	Provides planning and deployment solutions for OpenSSO Enterprise.
Sun OpenSSO Enterprise 8.0 Administration Guide	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).

TABLE P–1 OpenSSO Enterprise Documentation Set *(Continued)*

Title	Description
Sun OpenSSO Enterprise 8.0 Administration Reference	Provides reference information for the OpenSSO Enterprise command-line interface (CLI), configuration attributes, log files, and error codes.
Sun OpenSSO Enterprise 8.0 Developer's Guide	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.
Sun OpenSSO Enterprise 8.0 C API Reference for Application and Web Policy Agent Developers	Provides summaries of data types, structures, and functions that make up the public OpenSSO Enterprise C APIs.
Sun OpenSSO Enterprise 8.0 Java API Reference	Provides information about the implementation of Java packages in OpenSSO Enterprise.
Sun OpenSSO Enterprise 8.0 Upgrade Guide	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.
Sun OpenSSO Enterprise 8.0 Performance Tuning Guide	Provides information about how to tune OpenSSO Enterprise and its related components for optimal performance.

Policy Agent Documentation

Policy agent documentation includes these collections:

- Policy Agent 3.0: `http://docs.sun.com/coll/1767.1`
- Policy Agent 2.2: `http://docs.sun.com/coll/1322.1`

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`

TABLE P–2 Related Product Documentation *(Continued)*

Product	Link
Sun Java System Web Proxy Server 4.0.6	`http://docs.sun.com/coll/1311.6`
Sun Identity Manager 8.0	`http://docs.sun.com/coll/1514.5`

Searching Sun Product Documentation

Besides searching Sun product documentation from the docs.sun.com 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

See the following web sites for additional resources:

- Documentation (`http://docs.sun.com`)
- Support (`http://www.oracle.com/us/support/systems/index.html`)
- Training (`http://education.oracle.com`) – Click the Sun link in the left navigation bar.

Oracle Welcomes Your Comments

Oracle welcomes your comments and suggestions on the quality and usefulness of its documentation. If you find any errors or have any other suggestions for improvement, go to `http://docs.sun.com` and click Feedback. Indicate the title and part number of the documentation along with the chapter, section, and page number, if available. Please let us know if you want a reply.

Oracle Technology Network (`http://www.oracle.com/technetwork/index.html`) offers a range of resources related to Oracle software:

- Discuss technical problems and solutions on the Discussion Forums (`http://forums.oracle.com`).
- Get hands-on step-by-step tutorials with Oracle By Example (`http://www.oracle.com/technology/obe/start/index.html`).
- Download Sample Code (`http://www.oracle.com/technology/sample_code/index.html`).

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 `ls -a` to list all files. `machine_name% you have mail.`

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

aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* 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 shells that are included in the Oracle Solaris OS. Note that the default system prompt that is displayed in command examples varies, depending on the Oracle Solaris release.

TABLE P–4 Shell Prompts

Shell	Prompt
Bash shell, Korn shell, and Bourne shell	`$`
Bash shell, Korn shell, and Bourne shell for superuser	`#`
C shell	`machine_name%`
C shell for superuser	`machine_name#`

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
zip-root	Represents the directory where the `opensso_enterprise_80.zip` file is unzipped.

OpenSSO-Deploy-base	Represents the deployment directory where the web container deploys the `opensso.war` file.
	
	This value varies depending on the web container. To determine the value of *OpenSSO-Deploy-base*, 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:
	■ 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.
	
	The `.openssocfg` directory is in the `root` home directory (`/`), and the file name in `.openssocfg` is:
	
	`AMConfig_opt_SUNWappserver_domains_domain1_applications_j2ee-modules_opensso_`
	
	Then, the value for *OpenSSO-Deploy-base* is:
	
	`/opt/SUNWappserver/domains/domain1/applications/j2ee-modules/opensso`

TABLE P–5 Default Paths and Directory Names *(Continued)*

Term	Description
ConfigurationDirectory	Represents the name of the configuration directory specified during the initial configuration of OpenSSO Enterprise server instance using the Configurator. The default is `opensso` in the home directory of the user running the Configurator. Thus, if the Configurator is run by `root`, *ConfigurationDirectory* is `/opensso`.

Revision History

TABLE P–6 Revision History

Date (Version)	Description of Change
August 10, 2010 (16)	■ Revised “OpenSSO Enterprise Security Permissions for WebLogic Server” on page 37. ■ Revised outdated URLs.
May 6, 2010 (15)	■ Added Chapter 7, “Running the OpenSSO Diagnostic Tool.” ■ Revised Chapter 15, “Enabling the Access Manager SDK (AMSDK) Identity Repository Plug-in,” for CR 6949512.
September 10, 2009 (14)	■ Revised Chapter 19, “Using Active Directory as the User Data Store,” for issue 5052. ■ Revised Chapter 23, “Patching OpenSSO Enterprise 8.0,” for issue 5069.
June 16, 2009 (13)	■ Added the security policy file permissions for Apache Tomcat in “Adding Security Permissions For a Web Container” on page 34. ■ Added Chapter 22, “Taking Precautions Against Session-Cookie Hijacking in an OpenSSO Enterprise Deployment.”
April 17, 2009 (12)	■ Added the security policy file permissions for IBM WebSphere Application Server 6.1 in “Adding Security Permissions For a Web Container” on page 34. ■ Added the restriction that the OpenSSO configuration data store cannot be on an NFS-mounted file system in “OpenSSO Enterprise 8.0 Requirements” on page 19. ■ Resolved OpenSSO issues 4482 and 4514.
November 14, 2008 (11)	Updated for late changes.
November 11, 2008 (10)	Initial release.
August 6, 2008 (05)	Early Access (EA) release draft.

Oracle Welcomes Your Comments

Oracle 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.

CHAPTER 1

Getting Started With OpenSSO Enterprise 8.0

Sun OpenSSO Enterprise 8.0 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 following site:

`http://www.sun.com/software/products/opensso_enterprise`

Before you install and configure OpenSSO Enterprise:

- "OpenSSO Enterprise 8.0 Requirements" on page 19
- "Overview of Installing and Configuring OpenSSO Enterprise" on page 21
- "Using Sun Service Tags With OpenSSO Enterprise" on page 23

OpenSSO Enterprise 8.0 Requirements

TABLE 1–1 OpenSSO Enterprise 8.0 Requirements

Requirement	Description
File system	If you plan to use the OpenSSO configuration data store, you must deploy OpenSSO Enterprise on a local file system and not on an NFS-mounted file system. The OpenSSO configuration data store, which is deployed with OpenSSO Enterprise, is not supported on an NFS-mounted file system.

TABLE 1–1 OpenSSO Enterprise 8.0 Requirements *(Continued)*

Requirement	Description
Web container	One of the following web containers must be running on the host server where you plan to deploy OpenSSO Enterprise: ■ Sun Java System Application Server 9.1 Update 1 or Update 2 ■ GlassFish Application Server V2 UR1 or UR2 ■ Sun Java System Web Server 7.0 Update 3 ■ Apache Tomcat 6.0.18 (or later) ■ Oracle WebLogic Server 10 ■ Oracle WebLogic Server 9.2 MP2 ■ Oracle Application Server 10*g*, version 10.1.3.x ■ IBM WebSphere Application Server 6.1 ■ Apache Geronimo Application Server 2.1.2 (with Tomcat on Solaris systems only) ■ JBoss Application Server 4.x **Note**: These web container versions and any subsequent updates to the version are supported. For more information about supported versions and open issues for each web container, see the *Sun OpenSSO Enterprise 8.0 Release Notes*.
Configuration Data Store	OpenSSO Enterprise requires a data store for its configuration data, which you select when you run the GUI or command-line Configurator: ■ OpenSSO data store If you deploying OpenSSO Enterprise in a multiple server deployment, each OpenSSO Enterprise instance must share the same configuration data store. The OpenSSO configuration data store is not supported on an NFS-mounted file system. ■ Sun Java System Directory Server
User Data Store	OpenSSO Enterprise also requires a data store for its user data: ■ Sun Java System Directory Server If you are deploying multiple OpenSSO Enterprise instances in a multiple server deployment, all instances must access the same Directory Server. ■ Microsoft Active Directory ■ IBM Tivoli Directory Server ■ OpenSSO 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.

TABLE 1–1 OpenSSO Enterprise 8.0 Requirements *(Continued)*

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

Overview of Installing and Configuring OpenSSO Enterprise

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 the `opensso.war` file, using the web container administration console or deployment command. You no longer run a standalone installer.
- You initially configure OpenSSO Enterprise using the GUI or command-line 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.
- 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 data store (which is usually transparent to the user). OpenSSO Enterprise does not use the `AMConfig.properties` or `serverconfig.xml` files, except for co-existence with previous versions of Access Manager.

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. Check the *Sun OpenSSO Enterprise 8.0 Release Notes* for any recent issues or updates to the release.
2. If necessary, install, configure, and start one of the supported web containers listed in Table 1–1.
3. Download and unzip the `opensso_enterprise_80.zip` file from the following site:

   `http://www.oracle.com/technetwork/indexes/downloads/index.html`

   OpenSSO Enterprise 8.0 patch releases are available as patch ID 141655 on `http://sunsolve.sun.com/`.

   For information about installing a patch release, see Chapter 23, "Patching OpenSSO Enterprise 8.0."
4. Deploy the `opensso.war` file to the web container, using the web container administration console or deployment command.

   For the detailed steps, see Chapter 3, "Installing OpenSSO Enterprise."
5. Run either the GUI 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 4, "Configuring OpenSSO Enterprise Using the GUI Configurator," or Chapter 5, "Configuring OpenSSO Enterprise Using the Command-Line Configurator."

6. Launch OpenSSO Enterprise using the specific web container console or deployment command, or by specifying the URL from Step 4 in your browser.
7. Login to the Console as the OpenSSO Enterprise administrator (`amAdmin`) using the password you specified when you ran the Configurator.
8. 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*.
9. Depending on your security requirements, consider making a snapshot of your deployment using the OpenSSO Diagnostic Tool. Then, you can run the Tamper Detection test periodically to very the integrity of your deployment. For more information, see Chapter 7, "Running the OpenSSO Diagnostic Tool."

Using Sun Service Tags With OpenSSO Enterprise

OpenSSO Enterprise 8.0 is Service Tag enabled. To use Service Tags, you must first register your product. On the OpenSSO Enterprise Administration Console, under `Common Tasks`, click `Register This Product`.

To register, you need a Sun Online Account (SOA) or Sun Developer Network (SDN) account. If you do not have one of these accounts, you can get an account during the product registration process.

For more information about Sun Service Tags and Sun Connection, see `http://www.sun.com/service/sunconnection/index.jsp`.

CHAPTER 2

Deploying the OpenSSO Enterprise Web Container

Before you can deploy the Sun OpenSSO Enterprise `opensso.war` file, one of the following web containers must be installed, running, and configured on the host server. This chapter describes the considerations and deployment tasks (if any) for these web containers:

For more information, see also the "Web Containers Supported For OpenSSO Enterprise 8.0" in *Sun OpenSSO Enterprise 8.0 Release Notes*.

Planning Your OpenSSO Enterprise Web Container Deployment

Use the following table to plan your OpenSSO Enterprise web container deployment and configuration. For more detailed information, click the link for a each web container.

TABLE 2–1 OpenSSO Enterprise Web Containers

Web Container and Supported Versions	Required JVM Options	Required Java Permissions	OpenSSO Enterprise Pre-Deployment Tasks
"Sun Java System Application Server 9.1 Update 1 and Update 2" on page 27	Yes	Yes, if Java Security Manager is enabled: `server.policy`	Yes
"GlassFish Application Server V2 UR1 and UR2" on page 27	Yes	Yes, if Java Security Manager is enabled: `server.policy`	Yes
"Sun Java System Web Server 7.0 Update 3" on page 28	Yes	No	Yes
"Apache Tomcat 5.5.27 and 6.0.x" on page 28	Yes	Yes, if Java Security Manager is enabled: `catalina.policy`	Yes
"Oracle WebLogic Server 9.2 MP2" on page 29	Yes	Yes, if Java Security Manager is enabled: `weblogic.policy`	Yes
"Oracle WebLogic Server 10" on page 30	Yes	Yes, if Java Security Manager is enabled: `weblogic.policy`	Yes
"Oracle Application Server 10g" on page 30	Yes	Yes, if Security Manager for OC4J is enabled : `java2.policy`	No
"IBM WebSphere Application Server 6.1" on page 31	Yes	Yes, if Java Security Manager is enabled: `server.policy`	Yes
"Apache Geronimo Application Server 2.1.1" on page 33	Yes	Yes, if Java Security Manager is enabled: `geronimo.policy`	Yes
"JBoss Application Server 4.x" on page 34	Yes	Yes, if Java Security Manager is enabled: `server.policy`	Yes

Sun Java System Application Server 9.1 Update 1 and Update 2

Download location: `http://www.oracle.com/technetwork/indexes/downloads/index.html`

For the platforms that are supported for this web container, see "Platforms Supported For OpenSSO Enterprise 8.0" in *Sun OpenSSO Enterprise 8.0 Release Notes*.

OpenSSO Enterprise Pre-Deployment Tasks

1. In the Application Server 9.1 domain where you plan to deploy OpenSSO Enterprise server, change the following JVM options either using the Application Server admin console or command-line utility:
   - Change `-Xmx512m` to `-Xmx1024m`.
   - If the `-client jvm-option` is set, change it to `-server`.
2. If the Java Security Manager is enabled:
   - Set the following JVM option:

     ```
     -Dcom.sun.enterprise.server.ss.ASQuickStartup=false
     ```
   - Add the security permissions to the `server.policy` file, as described in "Adding Security Permissions For a Web Container" on page 34. After you edit the file, restart the web container.

GlassFish Application Server V2 UR1 and UR2

GlassFish site: `https://glassfish.dev.java.net/`

For the platforms that are supported for this web container, see "Platforms Supported For OpenSSO Enterprise 8.0" in *Sun OpenSSO Enterprise 8.0 Release Notes*.

Download locations:

- GlassFish V2 UR1: `https://glassfish.dev.java.net/downloads/v2ur1-b09d.html`
- GlassFish V2 UR2: `https://glassfish.dev.java.net/downloads/v2ur2-b04.html`

OpenSSO Enterprise Pre-Deployment Tasks

1. In the GlassFish domain where you plan to deploy OpenSSO Enterprise server, change the following JVM options either using the GlassFish administration console or by editing the `domain.xml` file:

- Change `-client` to `-server`.
- Change `-Xmx512m` to `-Xmx1024m`.

2. If the Java Security Manager is enabled:
   - Set the following JVM option:

     ```
     -Dcom.sun.enterprise.server.ss.ASQuickStartup=false
     ```
   - Add the security permissions to the `server.policy` file, as described in "Adding Security Permissions For a Web Container" on page 34.

     After you edit the file, restart the web container.

Sun Java System Web Server 7.0 Update 3

For the platforms that are supported for this web container, see "Platforms Supported For OpenSSO Enterprise 8.0" in *Sun OpenSSO Enterprise 8.0 Release Notes*.

Download location: `http://www.oracle.com/technetwork/indexes/downloads/index.html`

OpenSSO Enterprise supports Web Server 7.0 Update 3 only. Web Server 7.0 Update 1 and Web Server 7.0 Update 2 are **not** supported.

Web Server 7.0 Update 3 Documentation Center in the following collection: `http://docs.sun.com/coll/1653.3`

OpenSSO Enterprise Pre-Deployment Tasks

Using the Web Server 7.0 administration console or CLI, set the JVM heap size option from the default `-Xms128M -Xmx256M` to `-Xms256M -Xmx512M`.

Apache Tomcat 5.5.27 and 6.0.x

OpenSSO Enterprise supports Tomcat 5.5.27 or 6.0.x.

For the platforms that are supported for this web container, see "Platforms Supported For OpenSSO Enterprise 8.0" in *Sun OpenSSO Enterprise 8.0 Release Notes*.

Add the security permissions to the `catalina.policy` file, as described in "Adding Security Permissions For a Web Container" on page 34. After you edit the file, restart the web container.

For general information about Apache Tomcat, see `http://tomcat.apache.org/`.

OpenSSO Enterprise Pre-Deployment Tasks

1. Set the `-Xmx` JVM option to `-Xmx1024m`.
2. Add the `-Dcom.iplanet.am.cookie.c66Encode=true` JVM option to the `JAVA_OPTS` variable in the Tomcat `catalina.sh` or `catalina.bat` script. For example, for `catalina.sh`:

   ```
   if [ -r "$CATALINA_HOME"/bin/tomcat-juli.jar ]; then
   JAVA_OPTS="$JAVA_OPTS
   -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
   -Dcom.iplanet.am.cookie.c66Encode=true"
   ```

OpenSSO Enterprise Post-Deployment Tasks

After you deploy OpenSSO Enterprise on Tomcat, use the `ssoadm` utility to set the cookie encoding property to true. For example:

```
# ./ssoadm update-server-cfg \
-s http://openssohost.example.com:8080/opensso -u amadmin -f pwfile \
-a com.iplanet.am.cookie.encode=true
```

In this example, `pwfile` contains the password for `amadmin`.

Oracle WebLogic Server 9.2 MP2

For the platforms that are supported for this web container, see "Platforms Supported For OpenSSO Enterprise 8.0" in *Sun OpenSSO Enterprise 8.0 Release Notes*.

OpenSSO Enterprise Pre-Deployment Tasks

1. Set the `MaxPermSize` JVM option to a minimum value of 256 MB. For example:

   `-XX:MaxPermSize=256M`
2. If the Java Security Manager is enabled, add the security permissions to the `weblogic.policy` file, as described in "Adding Security Permissions For a Web Container" on page 34. After you edit the file, restart the web container.
3. See the following issues in the *OpenSSO Enterprise 8.0 Release Notes*:
   - "4077: OpenSSO Enterprise configuration on WebLogic Server requires new ldapjdk.jar" in *Sun OpenSSO Enterprise 8.0 Release Notes*.
   - "WebLogic Server StuckThreadMaxTime value is exceeded during configuration" in *Sun OpenSSO Enterprise 8.0 Release Notes*

Oracle WebLogic Server 10

For the platforms that are supported for this web container, see "Platforms Supported For OpenSSO Enterprise 8.0" in *Sun OpenSSO Enterprise 8.0 Release Notes*.

OpenSSO Enterprise Pre-Deployment Tasks

For the platforms that are supported for this web container, see "Platforms Supported For OpenSSO Enterprise 8.0" in *Sun OpenSSO Enterprise 8.0 Release Notes*.

1. Set the `MaxPermSize` JVM option to a minimum value of 256 MB. For example:

   ```
   -XX:MaxPermSize=256M
   ```
2. If the Java Security Manager is enabled, add the security permissions to the `weblogic.policy` file, as described in "Adding Security Permissions For a Web Container" on page 34. After you edit the file, restart the web container.
3. See the following issues in the *OpenSSO Enterprise 8.0 Release Notes*:
   - "4077: OpenSSO Enterprise configuration on WebLogic Server requires new ldapjdk.jar" in *Sun OpenSSO Enterprise 8.0 Release Notes*.
   - "WebLogic Server StuckThreadMaxTime value is exceeded during configuration" in *Sun OpenSSO Enterprise 8.0 Release Notes*

Oracle Application Server 10g

Oracle Application Server 10g version 10.1.3.x is supported.

For the platforms that are supported for this web container, see "Platforms Supported For OpenSSO Enterprise 8.0" in *Sun OpenSSO Enterprise 8.0 Release Notes*.

Oracle site: `http://www.oracle.com/technology/products/database/oracle10g`

OpenSSO Enterprise Pre-Deployment Tasks

If the Security Manager for Oracle Containers for Java EE (OC4J) is enabled with the JVM option `-Djava.security.manager`, append the permissions shown in Example 2–6 to the *ORACLE_HOME*`/j2ee/home/config/java2.policy` file.

IBM WebSphere Application Server 6.1

WebSphere Application Server 6.1 is supported on Solaris, Linux, Windows, and IBM AIX 5.3 systems.

- "OpenSSO Enterprise Pre-Deployment Tasks" on page 31
- "Post-Deployment Tasks" on page 32

If the Java Security Manager is enabled, add the security permissions to the `server.policy` file, as described in "Adding Security Permissions For a Web Container" on page 34. After you edit the file, restart the web container.

OpenSSO Enterprise Pre-Deployment Tasks

- "Adding `GenericJvmArguments`" on page 31
- "Adding Security Permissions" on page 31
- "Running the JSP Compiler" on page 31

Adding `GenericJvmArguments`

Add the `genericJvmArguments` using the WebSphere Admin Console or by editing the `server.xml` file:

1. Open the following file:

   install_root`/IBM/WebSphere/AppServer/profiles/AppSrv01/`
   `config/cells/`*cell*`/nodes/`*node*`/servers/`*server*`/server.xml`
2. Find the `jvmEntries` element.
3. Add the following `genericJvmArguments` and save the file:

   ```
   genericJvmArguments="-DamCryptoDescriptor.provider=IBMJCE
   -DamKeyGenDescriptor.provider=IBMJCE"
   ```
4. Restart WebSphere 6.1 Application Server.

Adding Security Permissions

If the Java Security Manager is enabled, add the security permissions to the `server.policy` file, as described in "Adding Security Permissions For a Web Container" on page 34. After you edit the file, restart the web container.

Running the JSP Compiler

The OpenSSO Enterprise JSP files require JDK 1.5 (or later), but on WebSphere Application Server 6.1, the JDK source level for JSP files is set to JDK 1.3 by default.

To reset the JDK source level on WebSphere Application Server 6.1:

1. Open the `WEB-INF/ibm-web-ext.xmi` file.

   JSP engine configuration parameters are stored either in a web module's configuration directory or in a web module's binaries directory in the `WEB-INF/ibm-web-ext.xmi` file:

   - Configuration directory. For example:

     ```
     {WAS_ROOT}/profiles/profilename/config/cells/cellname/applications/
     enterpriseappname/deployments/deployedname/webmodulename/
     ```

   - Binaries directory, if an application was deployed into WebSphere Application Server with the flag "Use Binary Configuration" flag set to `true`. For example:

     ```
     {WAS_ROOT}/profiles/profilename/installedApps/nodename/
     enterpriseappname/webmodulename/
     ```

2. Delete the `compileWithAssert` parameter by either deleting the statement from the file or enclosing the statement with comment tags (`<!-- ... -->`).
3. Add the `jdkSourceLevel` parameter with the value of 15. For example:

   ```
   <jspAttributes xmi:id="JSPAttribute_1" name="jdkSourceLevel" value="15"/>
   ```

   Note: The integer `(_1)` in `JSPAttribute_1` must be unique within the file.
4. Save the `ibm-web-ext.xmi` file.
5. Restart WebSphere Application Server for the new value to take effect.

For more information about the `jdkSourceLevel` parameter as well as other JSP engine configuration parameters, see:

`http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/rweb_jspengine.html`

Post-Deployment Tasks

Using the `ssoadm` and `ampassword` Utilities

The `setup` script in `ssoAdminTools.zip` installs the utilities and scripts. For information, see Chapter 6, "Installing the OpenSSO Enterprise Utilities and Scripts."

1. Before you run the `setup` script to install the utilities and scripts, modify the `setup` script. Before `-cp ...` in the last line, insert:

   ```
   -D"amCryptoDescriptor.provider=IBMJCE"
   -D"amKeyGenDescriptor.provider=IBMJCE"
   ```

2. Before you run `ssoadm`, add the following items to the `ssoadm` script:

   - Add `xalan.jar` to the `classpath` after `openfedlib.jar`. For example:

     ```
     ${TOOLS_HOME}/lib/xalan.jar
     ```

- Add the following items before `com.sun.identity.cli.CommandManager`:

  ```
  -D"amKeyGenDescriptor.provider=IBMJCE"
  -D"amCryptoDescriptor.provider=IBMJCE"
  ```

3. Before you run `ampassword`, add the following items to the `ampassword` script before `com.iplanet.services.ldap.ServerConfigMgr`:

   ```
   -D"amCryptoDescriptor.provider=IBMJCE"
   -D"amKeyGenDescriptor.provider=IBMJCE"
   ```

Apache Geronimo Application Server 2.1.1

OpenSSO Enterprise server supports Geronimo Application Server 2.1.1 with Tomcat on Solaris systems only.

OpenSSO Enterprise Pre-Deployment Tasks

1. Modify the `/geronimo-tomcat6-jee5-2.0.2/bin/geronimo.sh` file by adding `-X:MaxPermSize=512M`, as shown in the following start block:

   ```
   elif [ "$1" = "start" ] ; then
   shift
   touch "$GERONIMO_OUT"
   $START_OS_CMD "$_RUNJAVA" $JAVA_OPTS $GERONIMO_OPTS \
   $JAVA_AGENT_OPTS \
   -Dorg.apache.geronimo.base.dir="$GERONIMO_BASE" \
   -Djava.endorsed.dirs="$ENDORSED_DIRS" \
   -Djava.io.tmpdir="$GERONIMO_TMPDIR" \
   -XX:MaxPermSize=512M \
   -jar "$GERONIMO_HOME"/bin/server.jar $LONG_OPT "$@" \
   >> $GERONIMO_OUT 2>&1 &
   echo "" echo "Geronimo started in background. PID: $!"
   if [ ! -z "$GERONIMO_PID" ]; then echo $! > $GERONIMO_PID
   fi
   ```

2. Provide a deployment plan file either inside or outside of the `opensso.war` file. If placed inside the `opensso.war` file, name the plan `geronimo-web.xml` and place the file in `WEB-INF` directory. If placed outside of the WAR file, the plan file can be named otherwise. Here is a sample plan file:

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <web-app xmlns="http://geronimo.apache.org/xml/ns/j2ee/web-1.2">
   <environment>
   <moduleId>
   <groupId>sun</groupId>
   <artifactId>opensso</artifactId>
   <version>8.0</version>
   <type>war</type>
   </moduleId>
   </environment>
   <context-root>/opensso1</context-root>
   </web-app>
   ```

In the above example, the WAR file is deployed at:

```
geronimo-tomcat6-jee5-2.0.2/repository/sun/opensso/8.0/opensso-8.0.war
```

The web application is deployed at *protocol*`://`*server*`:`*port*`/opensso1`. You can change the deployment plan depending on your deployment scenario.

Related Information:

- Geronimo console URL: *protocol*`://`*server*`:8080/console/portal/welcome`
- Default user name and password: `system/manager`
- To start the Geronimo server:`/geronimo-tomcat6-jee5-2.0.2/bin/geronimo.sh start`
- To stop the Geronimo server: `/geronimo-tomcat6-jee5-2.0.2/bin/geronimo.sh stop`

JBoss Application Server 4.x

OpenSSO Enterprise server supports the Single Archive or Exploded Deployment on JBoss Application Server 4.x.

For information see `http://www.jboss.com/`.

See also "Examples: Deploying OpenSSO Enterprise on JBoss Application Server" on page 51.

For the platforms that are supported for this web container, see "Platforms Supported For OpenSSO Enterprise 8.0" in *Sun OpenSSO Enterprise 8.0 Release Notes*.

OpenSSO Enterprise Pre-Deployment Tasks

If you are using the Security Token Service (STS), set the `MaxPermSize` JVM option to a minimum value of 128 MB. For example:

```
-XX:MaxPermSize=128M
```

Adding Security Permissions For a Web Container

If the Java Security Manager is enabled for a web container, add the security permissions to the to the appropriate security policy file:

- "Adding OpenSSO Enterprise Security Permissions" on page 35
- "OpenSSO Enterprise Security Permissions for Apache Tomcat " on page 36
- "OpenSSO Enterprise Security Permissions for WebLogic Server" on page 37
- "OpenSSO Enterprise Security Permissions for IBM WebSphere Application Server 6.1" on page 39

- "OpenSSO Enterprise Security Permissions for JBoss Application Server" on page 40
- "OpenSSO Enterprise Security Permissions for Oracle Application Server" on page 41
- "OpenSSO Enterprise Security Permissions for Geronimo Application Server" on page 42

The security policy file depends on the web container:

- `server.policy` for most web containers. See "Adding OpenSSO Enterprise Security Permissions" on page 35.
- `weblogic.policy` for WebLogic Server. See "OpenSSO Enterprise Security Permissions for WebLogic Server" on page 37.
- `java2.policy` for Oracle Application Server. See "OpenSSO Enterprise Security Permissions for Oracle Application Server" on page 41.
- `geronimo.policy` for Geronimo Application Server 2.1.1. See "OpenSSO Enterprise Security Permissions for Geronimo Application Server" on page 42.

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

After you add the security permissions, restart the web container.

Adding OpenSSO Enterprise Security Permissions

These security permissions apply to Sun Java System Application Server 9.1 Update 1 and Update 2, and GlassFish Application Server V2 UR1 and UR2.

Add these permissions to the `server.policy` file.

EXAMPLE 2–1 OpenSSO Enterprise Security Permissions

```
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";
```

EXAMPLE 2–1 OpenSSO Enterprise Security Permissions *(Continued)*

```
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 java.lang.RuntimePermission "accessDeclaredMembers";
permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
permission javax.security.auth.AuthPermission "getSubject";
permission javax.management.MBeanTrustPermission "register";
permission java.lang.management.ManagementPermission "monitor";
permission javax.management.MBeanServerPermission "createMBeanServer";
permission java.util.PropertyPermission "javax.xml.soap.MetaFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.MessageFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPConnectionFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPFactory", "write";
permission java.net.NetPermission "getProxySelector";
permission java.security.SecurityPermission "getProperty.authconfigprovider.factory";
permission java.security.SecurityPermission "setProperty.authconfigprovider.factory";
permission javax.security.auth.AuthPermission "doAsPrivileged";
permission javax.security.auth.AuthPermission "modifyPublicCredentials";
permission java.security.SecurityPermission "insertProvider.XMLDSig";
permission java.security.SecurityPermission "putProviderProperty.WSS_TRANSFORM";
permission java.security.SecurityPermission "insertProvider.WSS_TRANSFORM";
permission java.security.SecurityPermission "getProperty.ocsp.*";
};
```

OpenSSO Enterprise Security Permissions for Apache Tomcat

Add the following permissions to the Apache Tomcat `catalina.policy` file.

EXAMPLE 2–2 OpenSSO Enterprise Security Permissions for Apache Tomcat

```
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";
```

EXAMPLE 2–2 OpenSSO Enterprise Security Permissions for Apache Tomcat *(Continued)*

```
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 java.lang.RuntimePermission "accessDeclaredMembers";
permission java.lang.RuntimePermission "setContextClassLoader";
permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
permission javax.security.auth.AuthPermission "getSubject";
permission javax.management.MBeanTrustPermission "register";
permission javax.management.MBeanPermission "*" , "*" ;
permission java.lang.management.ManagementPermission "monitor";
permission javax.management.MBeanServerPermission "createMBeanServer";
permission java.util.PropertyPermission "javax.xml.soap.MetaFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.MessageFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPConnectionFactory",
"write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPFactory", "write";
permission java.net.NetPermission "getProxySelector";
permission java.security.SecurityPermission
"getProperty.authconfigprovider.factory";
permission java.security.SecurityPermission
"setProperty.authconfigprovider.factory";
permission javax.security.auth.AuthPermission "doAsPrivileged";
permission javax.security.auth.AuthPermission "modifyPublicCredentials";
permission java.security.SecurityPermission "insertProvider.XMLDSig";
permission java.security.SecurityPermission "putProviderProperty.WSS_TRANSFORM";
permission java.security.SecurityPermission "insertProvider.WSS_TRANSFORM";
permission java.security.SecurityPermission "getProperty.ocsp.*";
};
```

OpenSSO Enterprise Security Permissions for WebLogic Server

Add these permissions to the `weblogic.policy` file.

EXAMPLE 2–3 OpenSSO Enterprise Security Permissions for the WebLogic Server `weblogic.policy` File

```
grant {
permission java.net.SocketPermission "*", "listen,connect,accept,resolve";
permission java.util.PropertyPermission "*", "read, write";
```

EXAMPLE 2–3 OpenSSO Enterprise Security Permissions for the WebLogic Server `weblogic.policy` File *(Continued)*

```
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 java.lang.RuntimePermission "accessDeclaredMembers";
permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
permission javax.security.auth.AuthPermission "getSubject";
permission javax.management.MBeanTrustPermission "register";
permission java.lang.management.ManagementPermission "monitor";
permission javax.management.MBeanServerPermission "createMBeanServer";
permission java.util.PropertyPermission "javax.xml.soap.MetaFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.MessageFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPConnectionFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPFactory", "write";
permission java.net.NetPermission "getProxySelector";
permission java.security.SecurityPermission "getProperty.authconfigprovider.factory";
permission java.security.SecurityPermission "setProperty.authconfigprovider.factory";
permission javax.security.auth.AuthPermission "doAsPrivileged";
permission javax.security.auth.AuthPermission "modifyPublicCredentials";
permission java.security.SecurityPermission "insertProvider.XMLDSig";
permission java.security.SecurityPermission "putProviderProperty.WSS_TRANSFORM";
permission java.security.SecurityPermission "insertProvider.WSS_TRANSFORM";
permission javax.management.MBeanPermission "*", "queryMBeans";
permission java.lang.RuntimePermission "setContextClassLoader";
};
```

OpenSSO Enterprise Security Permissions for IBM WebSphere Application Server 6.1

Add these permissions to the `server.policy` file.

EXAMPLE 2–4 OpenSSO Enterprise Security Permissions for IBM WebSphere Application Server 6.1

```
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 java.lang.RuntimePermission "accessDeclaredMembers";
permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
permission javax.security.auth.AuthPermission "getSubject";
permission javax.management.MBeanTrustPermission "register";
permission java.lang.management.ManagementPermission "monitor";
permission javax.management.MBeanServerPermission "createMBeanServer";
permission java.util.PropertyPermission "javax.xml.soap.MetaFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.MessageFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPConnectionFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPFactory", "write";
permission java.net.NetPermission "getProxySelector";
permission java.security.SecurityPermission "getProperty.authconfigprovider.factory";
permission java.security.SecurityPermission "setProperty.authconfigprovider.factory";
permission javax.security.auth.AuthPermission "doAsPrivileged";
permission javax.security.auth.AuthPermission "modifyPublicCredentials";
permission java.security.SecurityPermission "insertProvider.XMLDSig";
permission java.security.SecurityPermission "putProviderProperty.WSS_TRANSFORM";
permission java.security.SecurityPermission "insertProvider.WSS_TRANSFORM";
permission java.security.SecurityPermission "getProperty.ocsp.*";
```

EXAMPLE 2–4 OpenSSO Enterprise Security Permissions for IBM WebSphere Application Server 6.1 *(Continued)*

```
permission java.lang.RuntimePermission "createClassLoader";
permission java.lang.RuntimePermission "getClassLoader";
permission java.lang.RuntimePermission "setContextClassLoader";
permission java.lang.RuntimePermission "setFactory";
permission java.lang.RuntimePermission "setIO";
permission java.lang.RuntimePermission "modifyThread";
permission java.lang.RuntimePermission "stopThread";
permission java.lang.RuntimePermission "getProtectionDomain";
permission java.lang.RuntimePermission "readFileDescriptor";
permission java.lang.RuntimePermission "writeFileDescriptor";
permission java.lang.RuntimePermission "loadLibrary.*";
permission java.lang.RuntimePermission "accessClassInPackage.*";
permission java.lang.RuntimePermission "defineClassInPackage.*";
permission java.lang.RuntimePermission "accessDeclaredMembers";
permission java.lang.RuntimePermission "queuePrintJob";
permission java.io.FilePermission "<<ALL FILES>>", "read,write,execute,delete";
permission java.util.PropertyPermission "*", "read,write";
permission com.ibm.oti.shared.SharedClassPermission "*", "read,write";
permission com.ibm.websphere.security.WebSphereRuntimePermission "getSSLConfig",
"read,write,execute,delete";
};
```

OpenSSO Enterprise Security Permissions for JBoss Application Server

Add these permissions to the `server.policy` file.

EXAMPLE 2–5 OpenSSO Enterprise Security Permissions for JBoss Application Server

```
grant {
permission java.net.SocketPermission "*", "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";
```

EXAMPLE 2–5 OpenSSO Enterprise Security Permissions for JBoss Application Server *(Continued)*

```
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 java.lang.RuntimePermission "accessDeclaredMembers";
permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
permission javax.security.auth.AuthPermission "getSubject";
permission javax.management.MBeanTrustPermission "register";
permission java.lang.management.ManagementPermission "monitor";
permission javax.management.MBeanServerPermission "createMBeanServer";
permission java.util.PropertyPermission "javax.xml.soap.MetaFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.MessageFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPConnectionFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPFactory", "write";
permission java.net.NetPermission "getProxySelector";
permission java.security.SecurityPermission "getProperty.authconfigprovider.factory";
permission java.security.SecurityPermission "setProperty.authconfigprovider.factory";
permission javax.security.auth.AuthPermission "doAsPrivileged";
permission javax.security.auth.AuthPermission "modifyPublicCredentials";
permission java.security.SecurityPermission "insertProvider.XMLDSig";
permission java.security.SecurityPermission "putProviderProperty.WSS_TRANSFORM";
permission java.security.SecurityPermission "insertProvider.WSS_TRANSFORM";
};
```

OpenSSO Enterprise Security Permissions for Oracle Application Server

Add these permissions to the `java2.policy` file.

EXAMPLE 2–6 OpenSSO Enterprise Security Permissions For the Oracle `java2.policy` File

```
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";
```

EXAMPLE 2–6 OpenSSO Enterprise Security Permissions For the Oracle `java2.policy` File *(Continued)*

```
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 java.lang.RuntimePermission "accessDeclaredMembers";
permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
permission javax.security.auth.AuthPermission "getSubject";
permission javax.management.MBeanTrustPermission "register";
permission java.lang.management.ManagementPermission "monitor";
permission javax.management.MBeanServerPermission "createMBeanServer";
permission java.util.PropertyPermission "javax.xml.soap.MetaFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.MessageFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPConnectionFactory",
"write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPFactory", "write";
permission java.net.NetPermission "getProxySelector";
permission java.security.SecurityPermission
"getProperty.authconfigprovider.factory";
permission java.security.SecurityPermission
"setProperty.authconfigprovider.factory";
permission javax.security.auth.AuthPermission "doAsPrivileged";
permission javax.security.auth.AuthPermission "modifyPublicCredentials";
permission java.security.SecurityPermission "insertProvider.XMLDSig";
permission java.security.SecurityPermission "putProviderProperty.WSS_TRANSFORM";
permission java.security.SecurityPermission "insertProvider.WSS_TRANSFORM";
permission oracle.oc4j.security.OC4JRuntimePermission "oracle.oc4j.OC4JOnly";
};
```

OpenSSO Enterprise Security Permissions for Geronimo Application Server

▼ To Enable the Java Security Manager for Geronimo Application Server

1 **Create a new security policy file named `geronimo.policy` in the following directory:**

geronimo_home`/bin`

Add the security permissions in the `geronimo.policy` file, as shown in Example 2–7.

2 **In the `geronimo.sh` script, add following two lines under the `start` block:**

```
-Djava.security.manager \
-Djava.security.policy=geronimo.policy \
```

For example, the `start` block will look like:

```
elif [ "$1" = "start" ] ; then
  shift
  touch "$GERONIMO_OUT"
  $START_OS_CMD "$_RUNJAVA" $JAVA_OPTS $GERONIMO_OPTS \
    $JAVA_AGENT_OPTS \
    -Dorg.apache.geronimo.base.dir="$GERONIMO_BASE" \
    -Djava.endorsed.dirs="$ENDORSED_DIRS" \
    -Djava.ext.dirs="$EXT_DIRS" \
    -Djava.io.tmpdir="$GERONIMO_TMPDIR" \
    -Djava.security.manager \
    -Djava.security.policy=geronimo.policy \
    -XX:MaxPermSize=512M \
    -jar "$GERONIMO_HOME"/bin/server.jar $LONG_OPT "$@" \
       $GERONIMO_OUT 2>&1 &
    echo ""
    echo "Geronimo started in background. PID: $!"
    if [ ! -z "$GERONIMO_PID" ]; then
      echo $! > $GERONIMO_PID
    fi
```

3 **Restart Geronimo Application Server.**

Example 2–7 OpenSSO Enterprise Security Permissions for Geronimo Application Server

```
// ----------------------------------------------------------------------------
// Permissions for Geronimo Application Server
// ----------------------------------------------------------------------------
// Geronimo gets all permissions
grant codeBase "file:${org.apache.geronimo.base.dir}/lib/-" {
permission java.security.AllPermission;
};

grant codeBase "file:${org.apache.geronimo.base.dir}/repository/-" {
permission java.security.AllPermission;
};

grant {
permission java.lang.RuntimePermission "shutdownHooks";
permission java.lang.RuntimePermission "getenv.*";
permission java.lang.RuntimePermission "getProtectionDomain";
permission java.lang.RuntimePermission "modifyThread";
permission java.lang.RuntimePermission "getClassLoader";
permission java.lang.RuntimePermission "createSecurityManager";

permission javax.management.MBeanServerPermission "findMBeanServer";
permission javax.security.auth.AuthPermission "setReadOnly";
permission java.security.SecurityPermission "setPolicy";
permission java.security.SecurityPermission "getPolicy";
permission java.security.SecurityPermission "createAccessControlContext";
permission java.security.SecurityPermission "getProperty.package.definition";
permission java.security.SecurityPermission "setProperty.package.definition";
permission java.security.SecurityPermission "getProperty.package.access";
permission java.security.SecurityPermission "setProperty.package.access";
permission org.apache.geronimo.security.GeronimoSecurityPermission "getContext";
permission org.apache.geronimo.security.GeronimoSecurityPermission "setContext";
permission org.apache.geronimo.security.GeronimoSecurityPermission "configure";

permission java.util.PropertyPermission "Xorg.apache.geronimo.gbean.NoProxy", "read";
permission java.util.PropertyPermission "Xorg.apache.geronimo.kernel.config.Marshaler", "read";
};

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 java.lang.RuntimePermission "accessDeclaredMembers";
permission java.lang.RuntimePermission "setContextClassLoader";
permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
permission javax.security.auth.AuthPermission "getSubject";
permission javax.management.MBeanTrustPermission "register";
permission javax.management.MBeanPermission "*" , "*" ;
permission java.lang.management.ManagementPermission "monitor";
permission javax.management.MBeanServerPermission "createMBeanServer";
permission java.util.PropertyPermission "javax.xml.soap.MetaFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.MessageFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPConnectionFactory", "write";
permission java.util.PropertyPermission "javax.xml.soap.SOAPFactory", "write";
permission java.net.NetPermission "getProxySelector";
permission java.security.SecurityPermission "getProperty.authconfigprovider.factory";
permission java.security.SecurityPermission "setProperty.authconfigprovider.factory";
permission javax.security.auth.AuthPermission "doAsPrivileged";
permission javax.security.auth.AuthPermission "modifyPublicCredentials";
permission java.security.SecurityPermission "insertProvider.XMLDSig";
permission java.security.SecurityPermission "putProviderProperty.WSS_TRANSFORM";
permission java.security.SecurityPermission "insertProvider.WSS_TRANSFORM";
permission java.security.SecurityPermission "getProperty.ocsp.*";
};
```

CHAPTER 3

Installing OpenSSO Enterprise

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

- "Downloading OpenSSO Enterprise" on page 47
- "Deploying the OpenSSO Enterprise WAR File" on page 49
- "Creating and Deploying Specialized OpenSSO Enterprise WAR Files" on page 51
- "Examples: Deploying OpenSSO Enterprise on JBoss Application Server" on page 51

Before you begin, check the "OpenSSO Enterprise 8.0 Requirements" on page 19.

Downloading OpenSSO Enterprise

OpenSSO Enterprise is available in the `opensso_enterprise_80.zip` file, which you can download from the following site:

`http://www.oracle.com/technetwork/indexes/downloads/index.html`

The following table describes the layout after you unzip the `opensso_enterprise_80.zip` file. The directory where you unzip the file is represented by *zip-root*.

TABLE 3–1 OpenSSO Enterprise `opensso_enterprise_80.zip` File Layout

zip-root/`opensso`/ **Directory**	**Description**
`deployable-war`	OpenSSO Enterprise WAR and related files: ■ `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 49. ■ `console` contains the additional files for deploying only the OpenSSO Enterprise Console. See Chapter 11, "Installing the OpenSSO Enterprise Console Only." ■ `distauth` contains the additional files for deploying a Distributed Authentication UI server. See Chapter 9, "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 10, "Deploying the Identity Provider (IDP) Discovery Service." ■ `noconsole` contains the additional files for deploying OpenSSO Enterprise server without the Console. See Chapter 12, "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 51.
`docs`	Java API reference documentation (`opensso-public-javadocs.jar`).
`integrations`	■ `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 3–1 OpenSSO Enterprise `opensso_enterprise_80.zip` File Layout *(Continued)*

zip-root/`opensso/` **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 future use.
`samples`	Client SDK and samples (`opensso-client.zip`). See Chapter 13, "Installing the OpenSSO Enterprise Client SDK."
`tools`	OpenSSO Enterprise tools and utilities: ■ `ssoAdminTools.zip` contains files to setup and run the OpenSSO Enterprise command-line (CLI) utilities and scripts such as `ssoadm` and `ampassword`. ■ `ssoSessionTools.zip` contains the files to setup and configure OpenSSO Enterprise session failover. ■ `helpers` contains files for the UNIX authentication helper (`amunixd`). See Chapter 6, "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`.

Deploying the OpenSSO Enterprise WAR File

Before you deploy the `opensso.war`, a supported web container must be deployed and configured, as described in Chapter 2, "Deploying the OpenSSO Enterprise Web Container."

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

Caution – If you plan to use the OpenSSO configuration data store, you must deploy OpenSSO Enterprise on a local file system and not on an NFS-mounted file system. The OpenSSO configuration data store, which is deployed with OpenSSO Enterprise, is not supported on an NFS-mounted file system.

▼ 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/pwdfile
--port 4848 zip-root/opensso/deployable-war/opensso.war
```

   where:
   - *zip-root* is where you unzipped the `opensso_enterprise_80.zip` file. Or, if you copied `opensso.war` to a different location, use that location in the command.
   - `/tmp/pwdfile` 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 4, "Configuring OpenSSO Enterprise Using the GUI Configurator"

or

Chapter 5, "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 9, "Deploying a Distributed Authentication UI Server"
- IDP Discovery Service: Chapter 10, "Deploying the Identity Provider (IDP) Discovery Service"
- OpenSSO Enterprise Administration Console only: Chapter 11, "Installing the OpenSSO Enterprise Console Only"
- OpenSSO Enterprise server without the Administration Console: Chapter 12, "Installing OpenSSO Enterprise Server Only"
- OpenSSO Enterprise client SDK: Chapter 13, "Installing the OpenSSO Enterprise Client SDK"

Examples: Deploying OpenSSO Enterprise on JBoss Application Server

This section describes two additional methods to deploy OpenSSO Enterprise. Each method uses JBoss Application Server as the web container, but you can also use these methods on other web containers, if the container support the method.

- "Method 1: Deploying OpenSSO Enterprise Server on JBoss Application Server Using the Exploded Archive Method" on page 51
- "Method 2: Deploing OpenSSO Enterprise Server on JBoss Application Server Using the Traditional Single Archive Method" on page 52

Method 1: Deploying OpenSSO Enterprise Server on JBoss Application Server Using the Exploded Archive Method

▼ To Deploy OpenSSO Enterprise Server on JBoss Application Server Using the Exploded Archive Method

1. **Create a subdirectory under** *JBOSS_HOME*`/server/`*instance*`/deploy/`*name_of_war_file*. **For example:**

```
# mkdir /opt/jboss-4.2.2.GA/server/opensso/deploy/opensso.war
```

2 **Explode the `opensso.war` file in this new directory. For example:**

```
# cd /opt/jboss-4.2.2.GA/server/opensso/deploy/opensso.war
# jar xvf /tmp/opensso.war
```

Your don't need to restart the container, because JBoss Application Server will automatically hot-deploy it.

3 **Point your browser to `http://`*host*`.`*domain*`:`*port*`/opensso` or `http://`*host*`:`*port*`/opensso`and start configuring OpenSSO Enterpriseserver.**

4 **The OpenSSO Enterprise Configurator will write a bootstrap file in your home directory. For example:**

```
/.openssocfg/AMConfig_opt_jboss-4.2.2.GA_server_opensso_._deploy_opensso.war_
```

Method 2: Deploing OpenSSO Enterprise Server on JBoss Application Server Using the Traditional Single Archive Method

▼ To Deploy OpenSSO Enterprise Server on JBoss Application Server Using the Traditional Single Archive Method

1 **Explode the `opensso.war` file in a temporary directory. For example:**

```
# cd /tmp/sun
# jar xvf zip-root/opensso/deployable-war/opensso.war
```

where *zip-root* is where you unzipped the `opensso_enterprise_80.zip` file.

2 **In the `WEB-INF/classes/bootstrap.properties` file, uncomment the `configuration.dir` property and set the property to the configuration directory that you plan to use for the OpenSSO Enterprise installation. For example:**

```
configuration.dir=/opt/opensso-server1
```

3 **Create a new `opensso.war` with the extracted contents. For example:**

```
# cd /tmp/sun
# jar cvf /tmp/opensso.war *
```

4 **Hot-deploy this new WAR file on the JBoss Application Server container instance by copying the `opensso.war` from Step 3 to the JBoss `deploy` directory. For example:**

```
# cp /tmp/opensso.war /opt/jboss-4.2.2.GA/server/opensso/deploy
```

5 **Start configuring OpenSSO Enterprise by pointing your browser to** `http://`*host*`.`*domain*`:`*port*`/opensso` **or** `http://`*host*`:`*port*`/opensso`**.**

Note: Because you pre-configured the OpenSSO Enterprise configuration directory in the `opensso.war` file in Step 4, you won't be able to change it during the configuration process.

CHAPTER 4

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 55
- "Configuring OpenSSO Enterprise With the Default Configuration" on page 57
- "Configuring OpenSSO Enterprise With a Custom Configuration" on page 58

To run the Configurator from the command-line, see Chapter 5, "Configuring OpenSSO Enterprise Using the Command-Line Configurator."

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.**

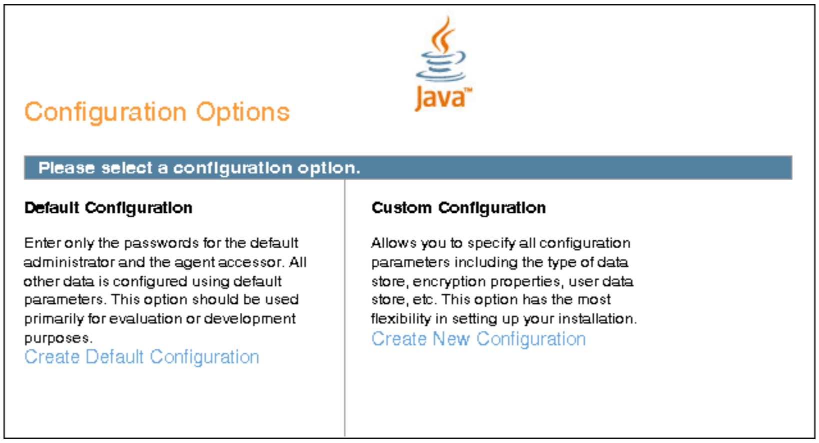
When you access OpenSSO Enterprise for the first time, you will be directed to the Configurator, to perform the OpenSSO Enterprise initial configuration.

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:



Configuration Options

Please select a configuration option.

Default Configuration

Enter only the passwords for the default administrator and the agent accessor. All other data is configured using default parameters. This option should be used primarily for evaluation or development purposes.
Create Default Configuration

Custom Configuration

Allows you to specify all configuration parameters including the type of data store, encryption properties, user data store, etc. This option has the most flexibility in setting up your installation.
Create New Configuration

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.

  The default policy agent user is also referred to as an application user. This user can connect to OpenSSO Enterprise server from a client such as the client SDK or a distributed authentication UI server.

  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 57.

  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 58.

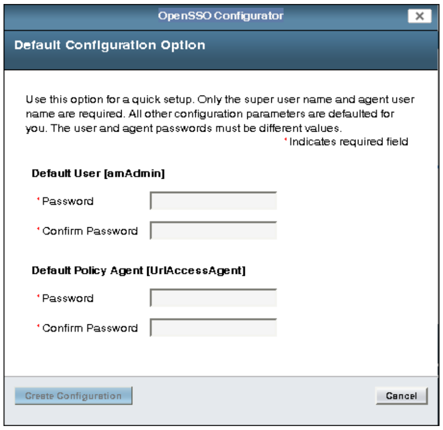
Configuring OpenSSO Enterprise With the Default Configuration

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.



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

The `amadmin` password must be at least 8 characters in length.

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 and the `install.log`, which if created, will be in the configuration directory (default `/opensso`). These logs might contain information about the cause of a configuration problem.

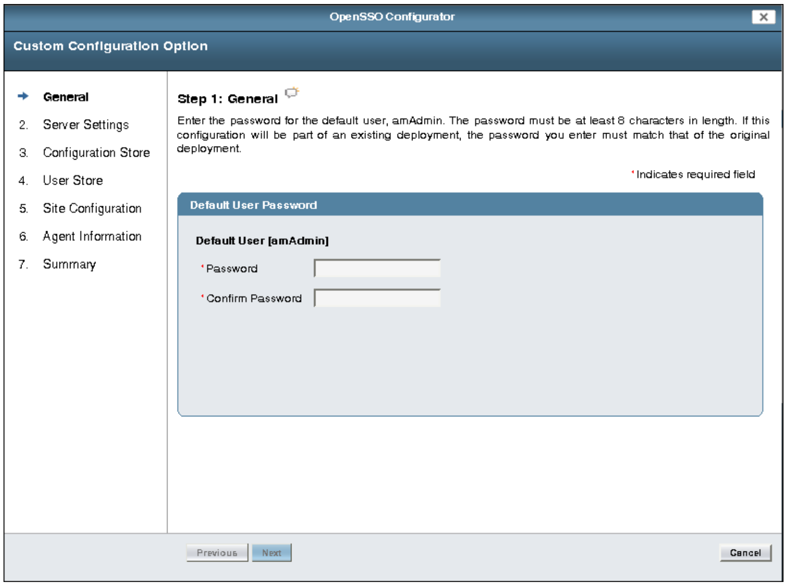
- Depending on your security requirements, consider making a snapshot of your deployment using the OpenSSO Diagnostic Tool. Then, you can run the Tamper Detection test periodically to very the integrity of your deployment. For more information, see Chapter 7, "Running the OpenSSO Diagnostic Tool."

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:



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]

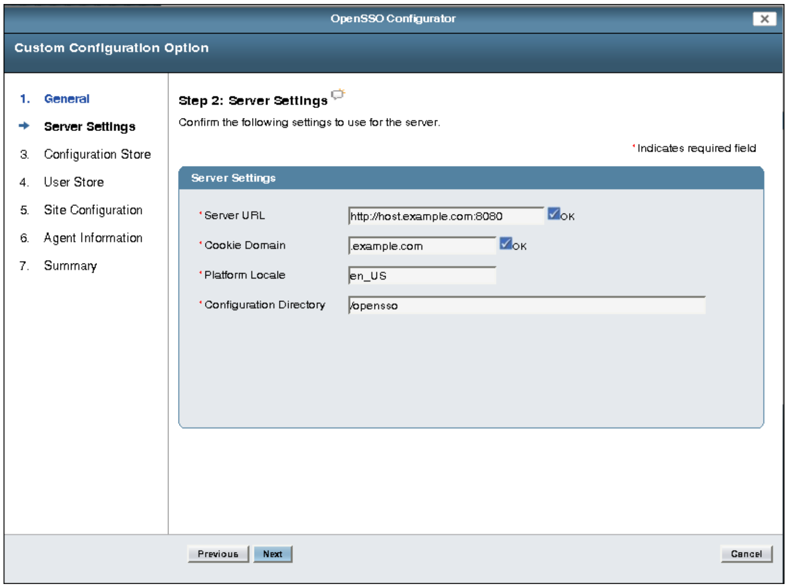
* Password

* Confirm Password

The `amadmin` password must be at least 8 characters in length.

Click **Next** to continue.

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



Step 2: Server Settings

Confirm the following settings to use for the server.

* Indicates required field

Server Settings

* Server URL	http://host.example.com:8080 ✓ OK
* Cookie Domain	.example.com ✓ OK
* Platform Locale	en_US
* Configuration Directory	/opensso

- **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_CN` (Simplified Chinese), or `zh_TW` (Traditional Chinese).
- **Configuration Directory** is the location of the OpenSSO Enterprise configuration directory.

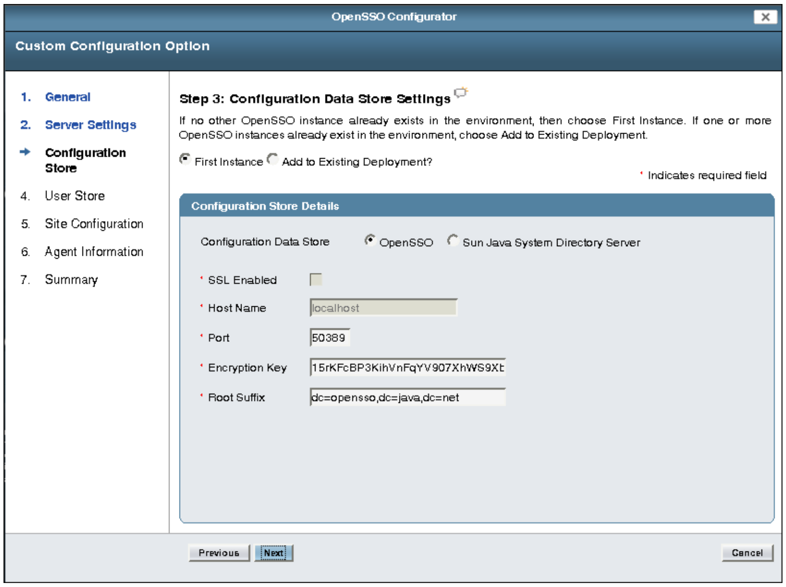
> **Important**: The runtime user of the OpenSSO Enterprise web container instance must have write access to the location where this directory will be created. For example, if the web container instance is running as the `webservd` user, then the `webservd` user must be able to write to the configuration 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** of the first already configured existing OpenSSO Enterprise server.



Step 3: Configuration Data Store Settings

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.

[✓ First Instance] Add to Existing Deployment?

* Indicates required field

Configuration Store Details

Configuration Data Store	[✓ OpenSSO] Sun Java System Directory Server
* SSL Enabled	☐
* Host Name	localhost
* Port	50389
* Encryption Key	15rKFcBP3KihVnFqYV907XhWS9Xt
* Root Suffix	dc=opensso,dc=java,dc=net

Configuration Store Details:

- **Configuration Data Store**:
  - **OpenSSO** stores OpenSSO Enterprise configuration data under the *configuration_directory*`/opends` directory on the local server.

    The OpenSSO configuration data store must be deployed on a local file system. It is not supported on an NFS-mounted file system.

    Replication. If you set up the OpenSSO configuration data store for replication, you must restart all OpenSSO Enterprise servers after you set up the replication. For more information, see Chapter 5, "Deploying and Configuring OpenSSO Enterprise," in *Deployment Example: Single Sign-On, Load Balancing and Failover Using Sun OpenSSO Enterprise 8.0*.

  - **Sun Java System Directory Server** stores OpenSSO Enterprise configuration data in Sun Java System Directory Server.
- **SSL Enabled**: Check if you want to enable SSL (`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.
- **Login ID** and **Password** are also required if you checked Sun Java System Directory Server.

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 *JAVA_HOME*`/jre/lib/security/cacerts` file, where *JAVA_HOME* is where you installed the JDK you are using.

For example, to import a root CA certificate to this key store:

```
keytool -keystore /usr/jdk/entsys-j2se/jre/lib/security/cacerts -keyalg RSA
-import -trustcacerts -alias "OpenSSO CA" -storepass changeit -file
/tmp/cacertfile.txt
```

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

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

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

You must also import the root CA certificate into the web container's JVM trust store for any instance that is attempting to connect to an `LDAPS`-enabled directory server.

Click **Next** to continue.

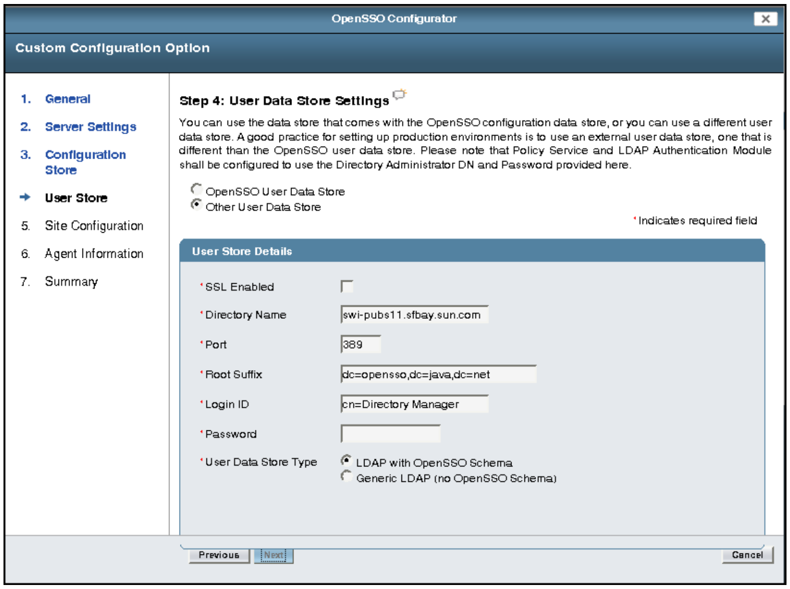
4 **Specify the User Data Store Settings:**

- **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.

  Multiple OpenSSO Enterprise instances. If you are configuring multiple OpenSSO Enterprise server instances to use the same Directory Server as the user data store, see Chapter 4, "Installing Sun Java System Directory Server and Creating Instances for Sun OpenSSO Enterprise User Data," in *Deployment Example: Single Sign-On, Load Balancing and Failover Using Sun OpenSSO Enterprise 8.0*.



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	swi-pubs11.sfbay.sun.com
* Port	389
* Root Suffix	dc=opensso,dc=java,dc=net
* Login ID	cn=Directory Manager
* Password	
* User Data Store Type	[✓ LDAP with OpenSSO Schema] Generic LDAP (no OpenSSO Schema)

User Store Details:

- **SSL Enabled**: Check if you want to enable SSL (`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 *JAVA_HOME*`/jre/lib/security/cacerts`. If this certificate is not imported, use the `keytool` utility to import the directory server root CA, where *JAVA_HOME* is where you installed the JDK you are using.

For example, to import a root CA certificate to this key store:

```
keytool -keystore /usr/jdk/entsys-j2se/jre/lib/security/cacerts -keyalg RSA
-import -trustcacerts -alias "OpenSSO CA" -storepass changeit -file
/tmp/cacertfile.txt
```

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

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

After you the import the certificate, restart the web container.

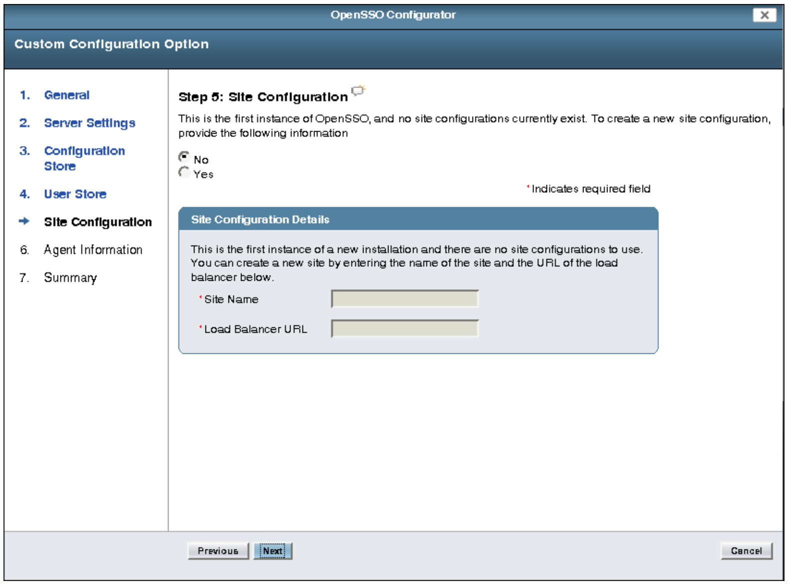
You must also import the root CA certificate into the web container's JVM trust store for any instance that is attempting to connect to an `LDAPS`-enabled directory server.

- **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 the `LDAPS` port of the directory server instance.
- **Root Suffix** is the user directory server initial or root suffix.
- **Login ID** is the administrator who has access to the user directory server.
- **Password** is the password for the user specified in Login ID.

  The Configurator automatically check the validity of this password.
- **User Data Store Type**:
  - **LDAP with OpenSSO Schema**: The directory server already has the OpenSSO Enterprise schema loaded. With this option, on a Sun Java System Directory Server instance, you can manage additional identity types such as roles and filtered roles as well as users and groups.
  - **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.


OpenSSO Configurator

Custom Configuration Option

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

Step 5: Site Configuration

This is the first instance of OpenSSO, and no site configurations currently exist. To create a new site configuration, provide the following information

[✓ No]
Yes

* Indicates required field

Site Configuration Details

This is the first instance of a new installation and there are no site configurations to use. You can create a new site by entering the name of the site and the URL of the load balancer below.

* Site Name

* Load Balancer URL

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.

Considerations about multiple OpenSSO Enterprise server instances:

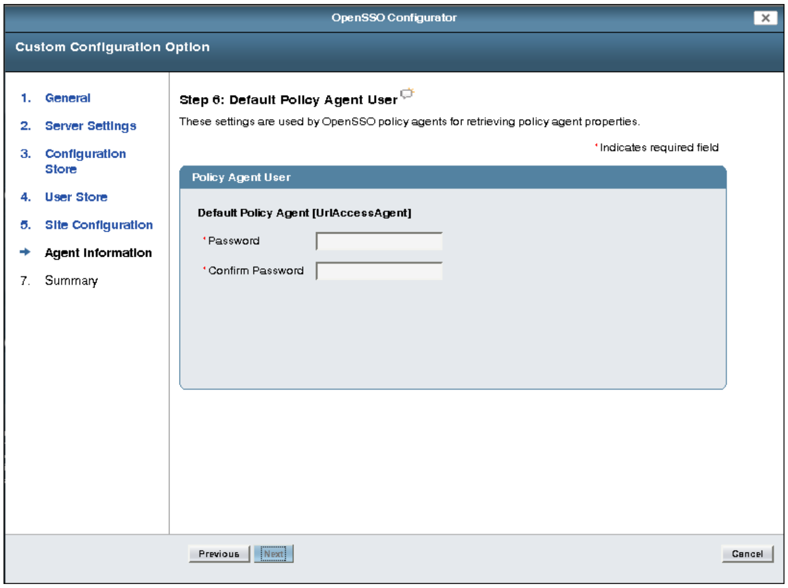
- **Multiple server instances as a site without stickiness**. For multiple OpenSSO Enterprise server instances deployed behind a load balancer without stickiness configured, to do additional configuration using the Admin Console, specify the URL of one of the OpenSSO Enterprise server instances and not the URL for the load balancer.

  If you are configuring an OpenSSO Server instance using `ssoadm`, see “Using `ssoadm` With OpenSSO Enterprise Configured as a Site” on page 77.

  For more information about configuring multiple OpenSSO Enterprise server instances as a site and using a load balancer, see Chapter 5, “Deploying and Configuring OpenSSO Enterprise,” in *Deployment Example: Single Sign-On, Load Balancing and Failover Using Sun OpenSSO Enterprise 8.0*.

- **Two instances not configured as a site**. If you are deploying two OpenSSO Enterprise server instances that share the same configuration data store but not configured as a site, you can log in to the Admin Console for first server instance and access the second server instance; however, after you configure the second server instance, you must restart the first server instance.

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



Step 6: Default Policy Agent User

These settings are used by OpenSSO policy agents for retrieving policy agent properties.

* Indicates required field

Policy Agent User

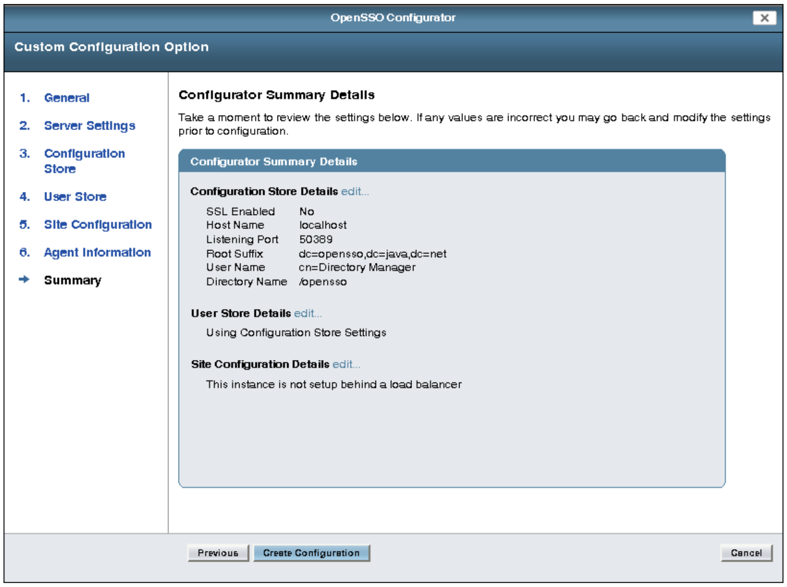
Default Policy Agent [UrlAccessAgent]

* Password

* Confirm Password

Click **Next** to continue.

7 Check the Summary page:



Configurator Summary Details

Take a moment to review the settings below. If any values are incorrect you may go back and modify the settings prior to configuration.

Configurator Summary Details

Configuration Store Details edit...

SSL Enabled	No
Host Name	localhost
Listening Port	50389
Root Suffix	dc=opensso,dc=java,dc=net
User Name	cn=Directory Manager
Directory Name	/opensso

User Store Details edit...

Using Configuration Store Settings

Site Configuration Details edit...

This instance is not setup behind a load balancer

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 configures 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.
- 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 and the `install.log`, which if created, will be in the configuration directory (default `/opensso`). These logs might contain information about the cause of a configuration problem.

- Depending on your security requirements, consider making a snapshot of your deployment using the OpenSSO Diagnostic Tool. Then, you can run the Tamper Detection test periodically to very the integrity of your deployment. For more information, see Chapter 7, "Running the OpenSSO Diagnostic Tool."

CHAPTER 5

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 have downloaded and unzipped the `opensso_enterprise_80.zip` file, as described in Chapter 3, "Installing OpenSSO Enterprise."
- You have deployed the `opensso.war` file in a supported web container, as described in "Deploying the OpenSSO Enterprise WAR File" on page 49.
- The web container must be started.
- Your `JAVA_HOME` environment variable must point to a JDK 1.5 or later.

Installing the Command-Line Configurator

After you unzip the `opensso_enterprise_80.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_enterprise_80.zip`.

▼ To Install the Command-Line Configurator

1. **Change to the** *zip-root*`/opensso/tools` **directory.**

2. **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. **Change to the directory where you unzipped the** `ssoConfiguratorTools.zip` **file.**

3. **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 Enteprise Configuration Parameters For the Command-Line Configurator" on page 71 for the properties you can set.

4. **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 Enteprise Configuration Parameters For the Command-Line Configurator

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

If `DATA_STORE=dirServer` is specified:

- The value for USERSTORE_TYPE under the "User Data Store Parameters" must be either `LDAPv3ForAMDS` or `LDAPv3`. The USERSTORE_TYPE cannot be blank or commented out.

  You must specify all of the relevant parameters for the user data store. For example:

  ```
  #Config Store Details
  DATA_STORE=dirServer
  DIRECTORY_SSL=SIMPLE
  DIRECTORY_SERVER=configurationdatastore.example.com
  DIRECTORY_PORT=5002
  ROOT_SUFFIX=dc=opensso,dc=java,dc=net
  DS_DIRMGRDN=cn=puser,ou=DSAME Users,dc=opensso,dc=java,dc=net
  DS_DIRMGRPASSWD=password

  # User Store Details
  USERSTORE_TYPE=LDAPv3ForAMDS
  USERSTORE_SSL=SIMPLE
  USERSTORE_HOST=userdatastore.example.com
  USERSTORE_PORT=5002
  USERSTORE_SUFFIX=dc=opensso,dc=java,dc=net
  USERSTORE_MGRDN=cn=puser,ou=DSAME Users,dc=opensso,dc=java,dc=net
  USERSTORE_PASSWD=password
  ```
- If 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 user who has read and write privileges to the root suffix and schema (`cn=schema`) in the configuration data store. Default: `DS_DIRMGRDN=cn=Directory Manager`
- DS_DIRMGRPASSWD is the password for the DS_DIRMGRDN user.

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:
  - `LDAPv3ForAMDS`: LDAP with OpenSSO Schema
  - `LDAPv3`: Generic LDAP (no OpenSSO Schema)
  - blank (`USERSTORE_TYPE=`): 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`

Note – Depending on your security requirements, consider making a snapshot of your deployment using the OpenSSO Diagnostic Tool. Then, you can run the Tamper Detection test periodically to very the integrity of your deployment. For more information, see Chapter 7, "Running the OpenSSO Diagnostic Tool."

CHAPTER 6

Installing the OpenSSO Enterprise Utilities and Scripts

The Sun OpenSSO Enterprise ZIP (`opensso_enterprise_80.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 76.
- `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 8, "Implementing OpenSSO Enterprise Session Failover."
- OpenSSO Enterprise 8.0 Update 1 Patch 4 and later releases includes `ssoDiagnosticTools.zip`, which contains the OpenSSO Diagnostic Tool. This tool allows you to run a number of diagnostic tests to verify configuration settings and to identify potential installation or deployment problems. For more information, see Chapter 7, "Running the OpenSSO Diagnostic Tool."

This chapter also describes:

- "Using `ssoadm` With OpenSSO Enterprise Configured as a Site" on page 77
- "Running the Unix Authentication Helper (`amunixd` Daemon)" on page 78

For information about uninstallation, see "Uninstalling the OpenSSO Enterprise Utilities and Scripts" on page 194

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

After you download and unzip the `opensso_enterprise_80.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 6–1 `ssoAdminTools.zip` File Layout

tools-zip-root **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, log, and debug directories. 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*.

Using `ssoadm` With OpenSSO Enterprise Configured as a Site

In a typical large deployment, OpenSSO Enterprise server instances are configured behind one or load balancers. The HTTP(s) traffic is usually one directional. That is, the traffic goes from one of the load balancers to the servers, but requests from servers are unable to reach the load balancers. If the above scenario applies to your deployment and you need to use the `ssoadm` utility (Solaris and Linux systems) or `ssoadm.bat` utility (Windows), perform the following procedure.

▼ To Use `ssoadm` With OpenSSO Enterprise Configured as a Site

1 **After you install the tools, edit the `ssoadm` or `ssoadm.bat` utility in the *tools-zip-root*/*deploy_uri*/`bin` directory.**

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`

2 **In the `ssoadm` or `ssoadm.bat` utility, add the following property to the `java` command:**

```
-D"com.iplanet.am.naming.map.site.to.server=
http://lb.example.com:58080/opensso=http://ssohost1.example.com:58080/opensso"
```

where:

- `lb` is the load balancer.
- `ssohost1` is the OpenSSO Enterprise server where `ssoadm` is installed.

3 **Save the `ssoadm` or `ssoadm.bat` utility.**

The utility can now send naming requests to the OpenSSO Enterprise server instance.

Once the site is enabled, this property prevents the administrator from being denied access to the server when the load balancer is inaccessible. When the `ssoadmin` command tries to access the load balancer, if the load balancer is not accessible, `ssoadmin` can directly access the server specified in this property.

Running the Unix Authentication Helper (amunixd Daemon)

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

After you unzip the `opensso_enterprise_80.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 (amunixd 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 `amunixd` daemon listens to at startup for configuration information. Default:`58946`
- **Authentication Port**: Port that the `amunixd` 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. For example, on some Linux systems, the PAM Service Name is `passwd`.If `password` or `passwd` is not correct, 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 `amunixd` daemon by running the `amunixd` script in the *zip-root*`/opensso/tools/helpers/bin` directory.**

For example:

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

Notes

- Run the `amunixd` 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 `amunixd` 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 `amunixd` process. For example:

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

  ```
  # ./amunixd -c portnumber
  ```
- If the you want the `amunixd` 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 ..."ipaddr`*N* are the IP addresses in the 3-dot (`111.111.111.111`) format of the systems that `amunixd` is to accept connections from.

CHAPTER 7

Running the OpenSSO Diagnostic Tool

OpenSSO Enterprise 8.0 Update 1 Patch 3 (and later) includes the OpenSSO Diagnostic Tool, which allows you to run a number of diagnostic tests to verify configuration settings and to identify potential installation or deployment problems. This chapter describes how to run the Tamper Detection test on your OpenSSO Enterprise configuration settings and server bits.

- "Getting Started With the OpenSSO Diagnostic Tool" on page 81
- "Running the Diagnostic Tool Tamper-Detection Tests" on page 82

Getting Started With the OpenSSO Diagnostic Tool

- "Unzipping the `ssoDiagnosticTools.zip` File" on page 81
- "Setting Your `JAVA_HOME` Environment Variable" on page 82
- "Invoking the Diagnostic Tool" on page 82

Unzipping the `ssoDiagnosticTools.zip` File

After you unzip the OpenSSO Enterprise 8.0 Update 1 Patch 4 (or later) ZIP file, the Diagnostic Tool is in the `ssoDiagnosticTools.zip` file in the *zip-root*`/opensso/tools` directory, where *zip-root* is where you unzipped the Patch 3 ZIP file.

Create a directory for the Diagnostic Tool and its related files and then unzip the `ssoDiagnosticTools.zip` file in the new directory. The tool is available for these platforms:

- Solaris and Linux systems: `ssodtool.sh`
- Windows: `ssodtool.bat`

In this guide, *diagnostic-tool-zip-root* represents the directory where you unzipped the `ssoDiagnosticTools.zip` file.

Setting Your `JAVA_HOME` Environment Variable

The Diagnostic Tool requires JDK 1.5 or later. Before you run the tool, set your `JAVA_HOME` environment variable to a JDK 1.5 or later installation.

Note: Embedded quotes are not supported in the `JAVA_HOME` environment variable.

Invoking the Diagnostic Tool

You can invoke the Diagnostic Tool in either CLI or GUI mode. First, change to the *diagnostic-tool-zip-root* directory and then invoke the tool as follows:

- GUI mode (default):
  - Solaris and Linux systems: `./ssodtool.sh`
  - Windows: `ssodtool.bat`
- CLI mode, using the `--console` option:
  - Solaris and Linux systems: `./ssodtool.sh --console`
  - Windows: `ssodtool.bat --console`

The default mode for the Diagnostic Tool is GUI. To change the default mode to CLI before you invoke the tool, set the following property in the *diagnostic-tool-zip-root*`/config/DTConfig.properties` configuration file:

```
odt.application.runmode=CLI
```

Running the Diagnostic Tool Tamper-Detection Tests

The Tamper-Detection test detects any changes made to the OpenSSO Enterprise server configuration settings or the deployed OpenSSO Enterprise server bits on the file system. To run the Tamper-Detection test, follow these general steps:

1. Run the Create Checksum test to create checksum files for the following:
   - OpenSSO Enterprise server configuration files and the OpenSSO Configuration Data Store

     Note. The Tamper Detection test does not report configuration changes made if your configuration data store is remote in Sun Java System Directory Server.
   - Deployed OpenSSO Enterprise server bits on the file system

   As a general guideline, run the Create Checksum tests after your OpenSSO Enterprise deployment is properly configured and functional. You must run the test twice, once for the configuration files and then again for the server bits.
2. Run the Detect Tamper test periodically to detect any changes made since the checksum files were created.

▼ To Run the Diagnostic Tool to Create Checksum Files

Before You Begin You must have unzipped the `ssoDiagnosticTools.zip` file and set your `JAVA_HOME` environment variable, as described in "Getting Started With the OpenSSO Diagnostic Tool" on page 81.

1. **Log in to the system where OpenSSO Enterprise is deployed and change to the directory where you unzipped the `ssoDiagnosticTools.zip` file.**

2. **Invoke the Diagnostic Tool. For example, in GUI mode on Solaris Systems:** `./ssodtool.sh`

3. **Under Category, select Tamper-Detection.**

4. **In Configuration Directory, specify one of the following paths:**

   - OpenSSO Enterprise server configuration path. For example: `/opensso`

     or

   - Web container directory path where the OpenSSO Enterprise server bits are deployed. For example, for Sun Java System Application Server 9.1:

     `/opt/SUNWappserver/domains/domain1/applications/j2ee-modules/opensso`

5. **Under Select Test, specify Create Checksum.**

6. **Click Run Selected.**

   The Diagnostic Tool creates a checksum file named _*configdir-pathname*`.checksum` in the *diagnostic-tool-zip-root*`/services/tamperdetection/backup` directory.

   For example:

   - `_opensso.checksum`
   - `_opt_SUNWappserver_domains_domain1_applications_j2ee-modules_opensso.checksum`

   Caution. The Tamper Detection test relies on the operating system security permissions to protect the checksum files. Depending on the requirements of your deployment, you might need to copy these files to a more secure location.

7. **To save the results as an HTML file, click Save All Results.**

8. **The Diagnostic Tool logs all test results in the *diagnostic-tool-zip-root*`/ssodtool.log` file. Optionally, as required by your deployment, check and save this log file.**

Next Steps Repeat this procedure to create a checksum file for the other path in Step 4.

▼ To Run the Diagnostic Tool Detect Tamper Test

Before You Begin You must have unzipped the `ssoDiagnosticTools.zip` file and set your `JAVA_HOME` environment variable, as described in "Getting Started With the OpenSSO Diagnostic Tool" on page 81.

Important: The Tamper Detection test expects the checksum files to be in the *diagnostic-tool-zip-root*`/services/tamperdetection/backup` directory, where they were generated when you ran the Create Checksum tests. Therefore, make sure that the checksum files are in this same directory before you run the Tamper Detection test.

1. **Log in to the system where OpenSSO Enterprise is deployed and change to the directory where you unzipped the `ssoDiagnosticTools.zip` file.**

2. **Invoke the Diagnostic Tool. For example, in GUI mode on Solaris Systems:** `./ssodtool.sh`

3. **Under Category, select Tamper-Detection.**

4. **In Configuration Directory, specify one of the following paths:**
   - OpenSSO Enterprise server configuration path. For example: `/opensso`

     or
   - Web container directory path where the OpenSSO Enterprise server bits are deployed. For example, for Sun Java System Application Server 9.1:

     `/opt/SUNWappserver/domains/domain1/applications/j2ee-modules/opensso`

5. **Under Select Test, specify Create Detect Tamper.**

6. **Click Run Selected.**

   The Diagnostic Tool uses the checksum file in the *diagnostic-tool-zip-root*`/services/tamperdetection/backup` directory to determine if any files have changed since the checksum file was created. The tool determines if a file:
   - Existed and was changed
   - Existed and was deleted.
   - Did not exist and was added

7. **To save the results as an HTML file, click Save All Results.**

8. **The Diagnostic Tool logs all test results in the** *diagnostic-tool-zip-root*`/ssodtool.log` **file. Optionally, as required by your deployment, check and save this log file.**

Next Steps Repeat this procedure to run the test for the other path in Step 4.

Examine the results to determine if your OpenSSO Enterprise deployment has been tampered with since you created the checksum files.

CHAPTER 8

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 87
- "Installing and Configuring the OpenSSO Enterprise Session Failover Components" on page 90
- "Configuring Session Failover in the OpenSSO Enterprise Console" on page 97

Overview of OpenSSO Enterprise Session Failover

- "OpenSSO Enterprise Session Failover Components" on page 87
- "OpenSSO Enterprise Session Failover Flow" on page 90

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 58.

    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.

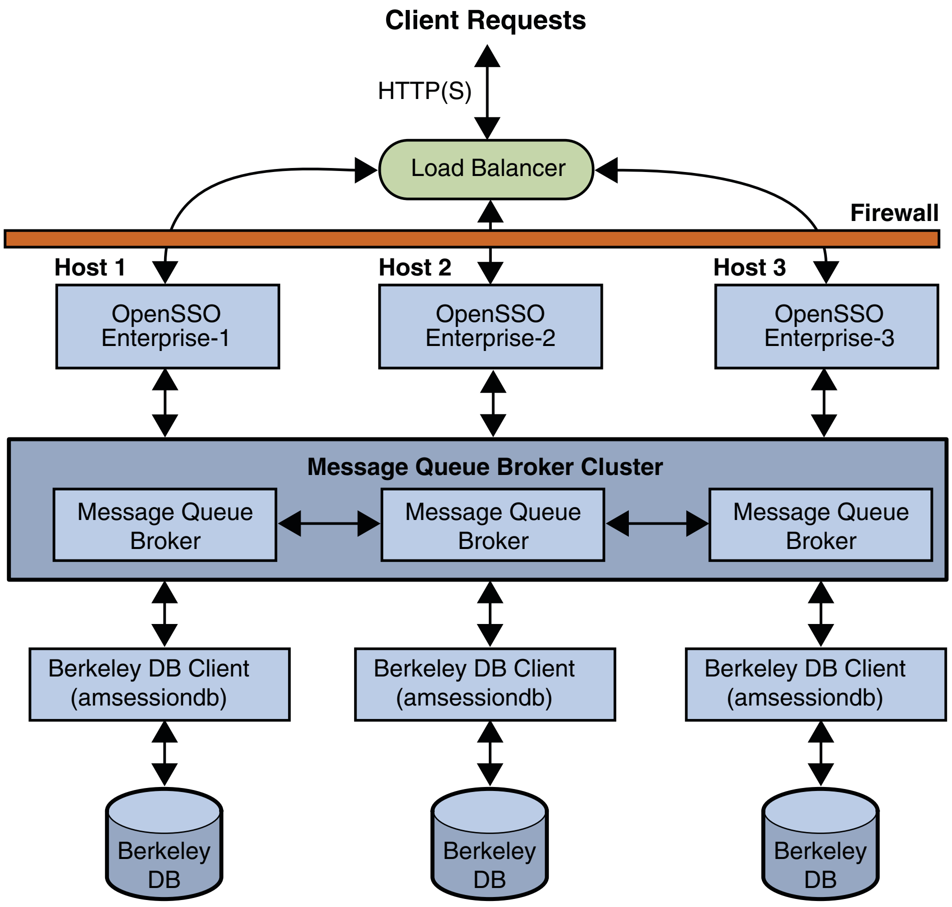
    or
  - OpenSSO data store: Instances must be configured for replication and act as a single directory server.

  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 8–1 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 91
- "Running the Session Failover `setup` Script" on page 92

- "Editing the `amsessiondb` Script (if Needed)" on page 93
- "Encrypting the Message Queue Broker Password Using the `amsfopassword` Script (Required)" on page 93
- "Running the `amsfo` Script to Start and Stop the Session Failover Components" on page 94

Unzipping the `ssoSessionTools.zip` File

The `ssoSessionTools.zip` file, which is part of the `opensso_enterprise_80.zip` file, contains the session failover scripts, JAR, properties, and related files.

▼ To Unzip the `ssoSessionTools.zip` File

Before You Begin Unzip the `opensso_enterprise_80.zip` file. The `ssoSessionTools.zip` file is then available in the *zip-root*`/opensso/tools` directory, where *zip-root* is where you unzipped `opensso_enterprise_80.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*.

sfo-zip-root File or Directory	Description
`README.txt`	Description of the `ssoSessionTools.zip` file.
`setup`	Script to install the session tools on Solaris and Linux systems.
`setup.bat`	Script to install the session tools on Windows systems.
`ext` directory	■ Message Queue JAR files for Solaris SPARC, Solaris x86, Linux. and Windows systems. ■ Oracle Berkeley DB JAR file (`je.jar`)
`lib` directory	■ JAR file for the `setup` scripts (`am_session_setup.jar`) ■ JAR file for the session API (`am_sessiondb.jar`)
`locale` directory	Properties file for the session API (`amSessionDB.properties`)
`template` 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:

sfo-zip-root **Directory**	**Script or File**
`jmq/mq`	Message Queue scripts and related files
dirname`/config/lib`	`amsfo.conf` session failover configuration file

sfo-zip-root **Directory**	**Script or File**
dirname/bin	■ Scripts to start and stop the Message Queue broker and `amsessiondb` client: ■ `amsfo` on Solaris and Linux systems ■ `amsfo.pl` on Windows systems ■ Scripts to run the Oracle Berkeley DB client (called by `amsfo`): ■ `amssessiondb` on Solaris and Linux systems ■ `amssessiondb.bat` on Windows systems ■ Scripts to encrypt the password for the Message Queue broker user (default is `guest`): ■ `amsfopassword` on Solaris and Linux systems ■ `amsfopassword.bat` on Windows systems

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 `amsfopassorwd` script accepts the password for the Message Queue broker user (default is `guest`) 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 clear-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 8–1.

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`, `jmq.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 8–1 `amsfo.conf` Configuration File Parameters

Variable	Description
`AM_HOME_DIR`	Specifies the following directory: *sfo-zip-root*/*dirname* where: ■ *sfo-zip-root* is where you unzipped the `ssoSessionTools.zip` file. ■ *dirname* is the name you specified when you ran the `setup` script to install the session failover scripts and related files.
`AM_SFO_RESTART`	Specifies (`true` or `false`) whether the script should automatically restart the Oracle Berkeley DB client (`amsessiondb`). The default is `true` (restart the `amsessiondb` client).

`CLUSTER_LIST`	Specifies the Message Queue broker list participating in the cluster. The format is:
	
	host1`:`*port*`,`*host2*`:`*port*`,` ... *hostn*`:`*port*
	
	For example:
	
	`mq1.example.com:7777,mq2.example.com:7777,`
	`mq3.example.com:7777`
	
	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.

`DATABASE_DIR`	Specifies the directory where the session database files will be created. Default: `/tmp/amsession/sessiondb`
`DELETE_DATABASE`	Specifies (`true` or `false`) whether the script should delete and then create a new database each time the Oracle Berkeley DB client (`amsessiondb`) is restarted. Default: `true`
`LOG_DIR`	Specifies the location of the log directory. Default: `/tmp/amsession/logs`
`START_BROKER`	Specifies (`true` or `false`) whether the Message Queue broker should be started with the `amsessiondb` process on the same server: `true` - The Message Queue broker will run on the same server as the `amsessiondb` process. `false` - The Message Queue broker and the `amsessiondb` process will run on different servers. Default: `true`

TABLE 8–1 `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`
`PASSWORDFILE`	Location of the password file that contains the encrypted password of the user name (default is `guest`) used to connect to the Message Queue broker. To generate the encrypted password, use the `amsfopassword` script, as described in "Encrypting the Message Queue Broker Password Using the `amsfopassword` Script (Required)" on page 93. Default: *sfo-zip-root*/*dirname*/`.password`
`AMSESSIONDB_ARGS`	`amsessiondb` script arguments. The `amsessiondb` script is called by the `amsfo` (or `amsfo.pl`) script. To determine the list of arguments, run: `amsession db -h`

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 8–1.

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.**

◆ ◆ ◆ CHAPTER 9

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:

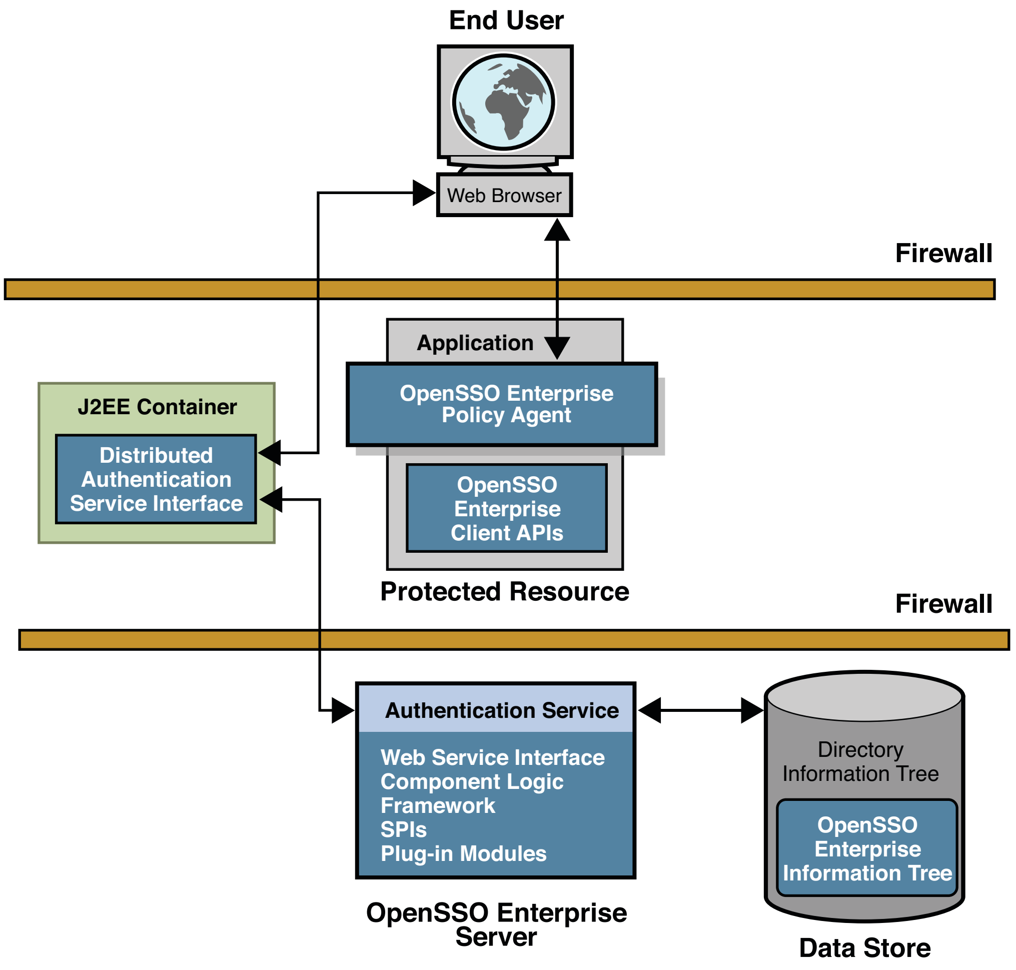
Distributed Authentication UI Server Overview

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 9–1 Distributed Authentication UI Server Deployment Scenario


End User
Web Browser
Firewall
Application
OpenSSO Enterprise Policy Agent
OpenSSO Enterprise Client APIs
Protected Resource
J2EE Container
Distributed Authentication Service Interface
Firewall
Authentication Service
Web Service Interface
Component Logic
Framework
SPIs
Plug-in Modules
OpenSSO Enterprise Server
Directory Information Tree
OpenSSO Enterprise Information Tree
Data Store

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 19.

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 Windows Desktop SSO and MSISDN authentication modules are not supported through the Distributed Authentication UI.

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_enterprise_80.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_enterprise_80.zip` file.

For more information about the `opensso.war` file, see "Downloading OpenSSO Enterprise" on page 47.

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 100 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.

Default values. The default values for the Distributed Authentication UI server protocol, host, port, and deployment URI will be based on the URL used to access the Distributed Authentication UI server WAR file. For example, if you use `http://distauth.example.com:8080/openssoDistAuth` to access the Configurator, the protocol will be `http`, the host will be `distauth.example.com`, the port will be `8080`, and the deployment URI will be `/openssoDistAuth`.

▼ 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-redi*
- 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).

◆ ◆ ◆ CHAPTER 10

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 107
- "Configuring the IDP Discovery Service" on page 108

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_enterprise_80.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_enterprise_80.zip` file.

For more information about the `opensso.war` file, see "Downloading OpenSSO Enterprise" on page 47.

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.

◆ ◆ ◆ CHAPTER 11

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

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 19.
- 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 196.

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_enterprise_80.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_enterprise_80.zip file.

For more information about the opensso.war file, see "Downloading OpenSSO Enterprise" on page 47.

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
```

CHAPTER 12

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 115
- "Generating a WAR File to Deploy OpenSSO Enterprise Server Only" on page 115
- "Deploying OpenSSO Enterprise Server Only" on page 116

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 19.

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_enterprise_80.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_enterprise_80.zip file.

For more information about the opensso.war file, see "Downloading OpenSSO Enterprise" on page 47.

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 4, "Configuring OpenSSO Enterprise Using the GUI Configurator"
- Chapter 5, "Configuring OpenSSO Enterprise Using the Command-Line Configurator"

◆ ◆ ◆ CHAPTER 13

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.

Contents

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 19.

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_enterprise_80.zip` file, as described in "Downloading OpenSSO Enterprise" on page 47.

  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*.

opensso-client-zip-root **Directory**	**Description**
`/sdk`	Client SDK CLI-based samples, which you can run in a standalone JVM outside of a web container: ▪ `/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: ▪ `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 Deploy these files using the web container administration console or command-line utility.

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 120.

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 your are using.

Also set your `PATH` correctly so that the appropriate JDK is used to compile the client SDK samples.

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*.

CHAPTER 14

Configuring OpenSSO Enterprise Sessions

Sun OpenSSO Enterprise session configuration includes:

- "Setting Session Quota Constraints" on page 123
- "Configuring Session Property Change Notifications" on page 127

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.

This section describes:

- "Deployment Scenarios for Session Quota Constraints" on page 123
- "Multiple Settings For Session Quotas" on page 124
- "Configuring Session Quota Constraints" on page 125

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 8, "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.

Note – By default, the COS priority for realm is set to medium, which is a value of 3 in OpenSSO Enterprise. The OpenSSO Console doesn't support changing the priority for realm-level service attributes. The Console supports only changing the priority for role-level service attributes. Therefore, in the OpenSSO Console, you can change the role priority to either higher or lower than the realm priority, to get the session attributes from the either the realm or role level.

▼ 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 127.

Read Timeout for Quota Constraint	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. Default: 6000 milliseconds

Resulting Behavior If Session Quota Exhausted	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: ■ `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. Default: `DESTROY_OLD_SESSION`
Exempt Top-Level Admins From Constraint Checking	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. Default: NO The super user defined for OpenSSO Enterprise (`com.sun.identity.authentication.super.user`) is always exempt from session quota constraint checking.
Deny User Login When Session Repository is Down	Specifies whether a user can login if the session repository is down. Takes effect only if the Enable Quota Constraints attribute is enabled. Default: NO
Maximum Session Time	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. Default: 120 minutes
Maximum Idle Time	Specifies the idle time in minutes before a session expires and the user must re-authenticate to regain access. Default: 30 minutes
Maximum Caching Time	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. Default: 3 minutes
Active User Sessions	Specifies the maximum number of concurrent sessions for a user. Default: 5

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`**.**

CHAPTER 15

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

Enabling the Access Manager SDK (AMSDK) Identity Repository (IdRepo) legacy plug-in allows Sun OpenSSO Enterprise to use the following features:

- Role-based authentication
- Role-based services

Contents

Requirements to Enable the AMSDK Identity Repository Plug-in

The requirements to enable and use the AMSDK Identity Repository plug-in 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 user data store.

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_enterprise_80.zip` file was unzipped.
- *config_dir* is the configuration directory specified during the initial configuration of `opensso.war`. For example: `/opensso`

Caution – Before you modify these LDIF files, be sure to back up each file.

Configure Directory Server by loading the required object classes and attributes by following one of these processes:

- "To Configure an Existing Directory Server With Access Manager 7.x User Data Store" on page 130
- "To Configure a New Directory Server" on page 131

▼ To Configure an Existing Directory Server With Access Manager 7.x User Data Store

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

- `sunFederationManagerDataStore`
- `sunFMSAML2NameIdentifier`

Note: The `fam_sds_schema.ldif` file also includes the `sunIdentityServerLibertyPPService` object class. If you don't want to load this object class, comment out the appropriate line before you load the file.

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

2 **Continue with "Configuring OpenSSO Enterprise Server" on page 132.**

▼ To Configure a New Directory Server

1 **In the following LDIF files, replace the tags marked by ampersands (@):**

- *config_dir*`/template/ldif/install.ldif`
  - `@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` (an occurrence for each user)
  - `@AMLDAPUSERPASSWD@` with the password for `amldapuser`
  - `@SERVER_HOST@` with the fully qualified host name. For example: `host.example.com`
  - `@ORG_OBJECT_CLASS@` with the organization object class. For example: `sunmanagedisorganization`
  - `@People_NM_ORG_ROOT_SUFFIX@` with the administrator for the people container (that is, the role that will manage the people container). For example: `opensso_dc=java_dc=net`
- *zip-root*`/opensso/ldif/index.ldif`
  - `@ORG_NAMING_ATTR@` with the organization naming attribute. For example: `o`
  - `@DB_NAME@` with the backend DB name. For example: `openssso`

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

You must configure OpenSSO Enterprise server for the AMSDK Identity Repository plug-in, using the `ssoadm` command. Consider these two scenarios to determine the steps you follow:

- **Scenario 1**: You do **not** want to customize the DAI service (`ums.xml` file). Follow "Configuring OpenSSO Enterprise Server Using the `ssoadm` Command with `add-amsdk-idrepo-plugin` Subcommand" on page 132.
- **Scenario 2**: You want to customize the DAI service (`ums.xml` file). Follow "Configuring OpenSSO Enterprise Server Manually" on page 133.

After you follow either scenario, continue with "Creating a Data Store Using the AMSDK Plug-in" on page 136.

Configuring OpenSSO Enterprise Server Using the `ssoadm` Command with `add-amsdk-idrepo-plugin` Subcommand

In this scenario, you do not want to customize the DAI service (`ums.xml` file). The `ssoadm` command with the `add-amsdk-idrepo-plugin` subcommand configures OpenSSO Enterprise server to enable the AMSDK Identity Repository plug-in 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 AMSDK Identity Repository plug-in

▼ To Configure OpenSSO Enterprise Server Using the `ssoadm` Command and `add-amsdk-idrepo-plugin` Subcommand

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

```
# ./ssoadm add-amsdk-idrepo-plugin -u amadmin -f ./password-file \
-a user-naming-attribute -o oranization-naming-attribute \
-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.

`-a` and `-o` specify the user naming attribute and organization naming attribute, respectively. Both parameters are optional. The default values are `uid` and `o`.

`-b` specifies the base `DN` of the Directory Server in which the Access Manager repository is being configured. 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 `proxyuser`.

On Solaris and Linux systems, the password files specified by `-x` and `-p` must have 400 (read-only by owner) permissions.

2 **Restart the OpenSSO Enterprise server web container.**

3 **Continue with "Creating a Data Store Using the AMSDK Plug-in" on page 136.**

Configuring OpenSSO Enterprise Server Manually

In this scenario, you want to customize the DAI service (`ums.xml` file), so you must configure OpenSSO Enterprise server manually by:

- "Loading the Directory Access Instructions (DAI) Service" on page 133
- "Loading the AMSDK Subschema" on page 134
- "Updating the Directory Server Information for the AMSDK Plug-in" on page 134
- "Enabling Persistent Search Connections for the AMSDK Plug-in" on page 135

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. For example, `uid` (which is the default)

- `@ORG_NAMING_ATTR@` with your organization naming attribute. For example, `o` (which is the default)

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_enterprise_80.zip` file was unzipped.

Loading the 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 zip-root/opensso/xml/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.

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 AMSDK Plug-in

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

▼ To Enable Persistent Search Connections for the AMSDK plug-in

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`**.**

c. **Click** `Save`**.**

4 **Log out of the Console.**

5 **Restart the OpenSSO Enterprise server web container.**

Creating a Data Store Using the AMSDK Plug-in

Use the following procedure to create a new data store or to verify that you correctly enabled the AMSDK Identity Repository plug-in.

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

1 **Log in to the OpenSSO Enterprise Administration Console as** `amadmin`**.**

2 **Click** `Access Control`**.**

3 **Under Realm Name, click the name of the realm.**

4 **Click** `Data Stores`**.**

5 **Click** `New`**.**

6 **For Select Type of Data Store, check** `Access Manager Identity Repository Plug-in`**.**

7 **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`.)

8 **If you are creating a new data store, provide the required configuration values.**

For other fields, either accept the default values or provide values appropriate for your deployment.

9 **Click** `Finish` **to complete the configuration.**

CHAPTER 16

Managing LDAP Persistent Searches

OpenSSO Enterprise can use LDAP persistent searches (psearches) to obtain asynchronous notifications of changes that occur in Sun Java System Directory Server. By default, however, persistent searches are not enabled for OpenSSO Enterprise.

Contents

Enabling Persistent Searches

The OpenSSO Enterprise Event Service (`amEventService`) creates and manages the persistent search connections. Since persistent searches are disabled by default, use the following methods to enable them

- `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.

▼ To Enable 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 only the searches you want to disable, including** `aci`**,** `sm`**, or** `um` **(or a combination, with each item separated by a comma).**

   That is, to enable a persistent search, make sure that `aci`, `sm`, or `um` is **not** present in this files.

4. **Click** `Save` **and log out of the Console.**

5. **Restart the OpenSSO Enterprise web container.**

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

You can also enable persistent searches by setting the `com.sun.am.event.connection.disable.list` property, using the `ssoadm` command.

To enable a specific persistent searches, make sure the respective persistent search value (`aci`, `sm`, or `um`) is **not** is not included in the property.

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

For example, to enable persistent searches for ACI changes only:

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

After you set the property, restart the OpenSSO Enterprise web container for the new values to take effect.

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 set 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

▼ To Disable Persistent Searches for a Data Store

1. **Log in to the Admin Console as** `amadmin`**.**
2. **Click** `Access Control`**,** *realm-name***,** `Data Stores`**,** *data-store-name***.**
3. **Set the** `Persistent Search Base DN` **field to blank.**
4. **Click** `Save` **and log out of the Console.**
5. **Restart the OpenSSO Enterprise web container.**

Disabling Persistent Searches on a Data Store

▼ To Disable Persistent Searches on a Data Store

1. **Log in to the Console as** `amadmin`**.**
2. **Click Access Control,** *realm-name***, Data Stores,** *data-store-name* **and then LDAPv3 Configuration**
3. **The "Persistent Search BaseDN" field must be empty (spaces).**
4. **If you made configuration changes, click Save.**

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 *Sun OpenSSO Enterprise 8.0 Developers Guide*.

◆ ◆ ◆ CHAPTER 17

Customizing OpenSSO Enterprise Administration Console Pages

For some deployments, you might need to customize Administration Console pages such as the login or logout 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_enterprise_80.zip`, as described in "Downloading OpenSSO Enterprise" on page 47.**

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_enterprise_80.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
`customwar/config/auth/default/Logout.jsp`	Console logout page

`customwar/config/auth/default/DataStore.xml`	Data store authentication page
`customwar/config/auth/default/LDAP.xml`	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 49.

CHAPTER 18

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 4, "Configuring OpenSSO Enterprise Using the GUI Configurator," or Chapter 5, "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_enterprise_80.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 Enteprise.

   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
```

   To determine the name of the backend database, use either of these methods:

   - Use the `ldapsearch` utility. For example:

     ```
     ldapsearch -h host-name -p port-number -s base -b "cn=config" -D \
     "cn=directory manager" -w password "objectclass=*" | grep backend
     ```

     ```
     nsslapd-backendconfig=cn=config,cn=red,cn=ldbm database,cn=plugins,cn=config
     ```

     In this example, the suffix is `dc=red,dc=iplanet,dc=com`, and the backend database name is `red`.

   - Check the `nsslapd-backend` property in the Directory Server instance's `dse.ldif` file.

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 a new file named `ldapentries`:

a. Add the entries shown in Example 18–1.

b. In the `ldapentries` file, replace:

- `dc=example,dc=com` with your root suffix
- *dsameuser-password* and *amldapuser-password* with your actual passwords

4 Load the entries in `ldapentries`. For example:

```
ldapmodify -h dshost -p dsport -D "cn=directory manager" \
-w dmpasswd -c -a -f ldapentries
```

5 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`.

Example 18–1 `ldapentries` File

```
dn: ou=people,dc=example,dc=com
objectClass: top
objectClass: organizationalUnit

dn: ou=agents,dc=example,dc=com
objectClass: top
objectClass: organizationalUnit
```

```
dn: ou=groups,dc=example,dc=com
objectClass: top
objectClass: organizationalUnit

dn: ou=dsame users,dc=example,dc=com
objectClass: top
objectClass: organizationalUnit

dn: cn=dsameuser,ou=DSAME Users,dc=example,dc=com
objectclass: inetuser
objectclass: organizationalperson
objectclass: person
objectclass: top
cn: dsameuser
sn: dsameuser
userPassword: dsameuser-password

dn: cn=amldapuser,ou=DSAME Users,dc=example,dc=com
objectclass: inetuser
objectclass: organizationalperson
objectclass: person
objectclass: top
cn: amldapuser
sn: amldapuser
userPassword: amldapuser-password

dn:dc=example,dc=com
changetype:modify
add:aci
aci: (target="ldap:///dc=example,dc=com")(targetattr="*") (version 3.0;
acl "S1IS special dsame user rights for all under the root suffix";
allow (all) userdn = "ldap:///cn=dsameuser,ou=DSAME Users,dc=example,dc=com"; )

dn:dc=example,dc=com
changetype:modify
add:aci
aci: (target="ldap:///dc=example,dc=com")(targetattr="*")(version 3.0;
acl "S1IS special ldap auth user rights";
allow (read,search) userdn = "ldap:///cn=amldapuser,ou=DSAME Users,dc=example,dc=com"; )
```

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

CHAPTER 19

Using Active Directory as the User Data Store

Sun OpenSSO Enterprise supports Microsoft Active Directory as the user data store.

Contents

Overview of Using Active Directory as the User Data Store

By default, OpenSSO Enterprise defines a set of object classes and attributes. These object classes and attributes are required in your Active Directory server if you want OpenSSO Enterprise to manage your Active Directory server.

The OpenSSO Console provides user management functionality based on the OpenSSO Enterprise predefined set of object classes and attributes, as specified through the OpenSSO Enterprise XML files. If the Active Directory server you are trying to access does not have these required object classes or the attributes defined, access involving the missing object class or attributes will fail, unless you change the user XML files to match the attributes defined for your Active Directory server.

For example, when you create a user via the OpenSSO Console, the Console writes out to the Active Directory server the predefined set of OpenSSO Enterprise object classes and attributes for the user. If the Active Directory server is not configured with the same set of user object classes and attributes, the user create operation will fail. When you use the Console's user information page to edit a user's information, unless the Active Directory server has the same set of attributes and/or object classes defined for the user as OpenSSO Enterprise does, the operation will fail.

The Access Manager Identity Repository (IdRepo) LDAPv3 plug-in provides attribute name mapping. You can refer to an attribute name as one name in OpenSSO Enterprise and a different name in your Active Directory server. As a result, you need not have all OpenSSO Enterprise attributes defined in Active Directory if you use attribute name mapping. However, if OpenSSO Enterprise has more attributes than you have in your Active Directory server, you cannot do one-to-one mapping, and some OpenSSO Enterprise read or write operations will fail due to missing attributes in the Active Directory server.

Requirements For Active Directory as the User Data Store

To configure and use Active Directory as the user data store, your deployment must meet these requirements:

- OpenSSO Enterprise 8.0 is installed on a supported web container.
- Active Directory 2003 is installed on Windows Server 2003 R2 with "Windows Server forest functional level" enabled. For more information, see:

  `http://support.microsoft.com/?id=322692#4`
- You have not made any changes to the OpenSSO Enterprise schema, attributes, or XML files.

Configuring Active Directory With the OpenSSO Enterprise Schema Files

The Access Manager Identity Repository (IdRepo) LDAPv3 plug-in must be able to assign the service's object class name to the user's object class attribute, so it can tell if a user has been assigned a given service. The following procedure describes how to load the OpenSSO Enterprise schema files into Active Directory and then to configure OpenSSO Enterprise to enable the OpenSSO Enterprise services.

▼ To Configure Active Directory with OpenSSO Enterprise Schema Files

1. **Back up the `am_remote_ad_schema.ldif` file.**

   After you have unzipped `opensso_enterprise_80.zip`, this file is available in the following directory:

   zip-root`/opensso/ldif`

2. **In the `am_remote_ad_schema.ldif` file, replace `@ROOT_SUFFIX@` with the root suffix of your Active Directory installation.**

3 **Using Active Directory tools (or another tool of your choice), load the `am_remote_ad_schema.ldif` file from the previous step into Active Directory.**

4 **Log in to the OpenSSO Administration Console. In the data store configuration page's LDAP User Attributes field, add the attribute names defined in the above LDIF file.**

5 **If you are writing your own service with dynamic user attributes, the `service.ldif` file for Active Directory must NOT have the following lines:**

```
dn: CN=User,CN=Schema,CN=Configuration,ROOT_SUFFIX
changetype: modify
add: auxiliaryClass
auxiliaryClass: yourClassname
```

Otherwise, OpenSSO Enterprise will not be able to assign the service's object class name to the user's object class attribute.

Configuring a Data Store For Active Directory

This section describes how to configure an Access Manager Identity Repository (IdRepo) LDAPv3 data store for Active Directory.

▼ To Configure a Data Store For Active Directory

1 **Log in to the OpenSSO Admin Console.**

2 **Click Access Control,** *realm-name***, Data Sores, and then New.**

3 **Enter the Name, check Active Directory, and then click Next.**

4 **Set the following Active Directory attributes.**

LDAP Server: Active Directory server name and port number that you want to connect to. For example: `myADServer.example.com:389`

LDAP Bind DN: `CN=Administrator,CN=Users,DC=example,DC=com`

LDAP Bind Password: Password for `CN=Administrator,CN=Users,DC=example,dc=com`

LDAP Organization DN: `DC=example,DC=com` — Organization `DN` that this datastore will map to. This will be the base `DN` of all operations performed in this data store.

Enable LDAP SSL: Select if the Active Directory server is in SSL mode.

LDAP Connection Pool Minimum Size: Initial number of connections in the connection pool. The use of connection pool avoids having to create a new connection each time.

LDAP Connection Pool Maximum Size: Maximum number of connections allowed.

Maximum Results Returned from Search: Maximum number of search results to return. This value should be based on the size of your LDAP organization. The maximum number returned cannot exceed the `ns` size limit configured for the Active Directory server.

Search Timeout: Maximum time in seconds to wait for results on a search operation.

LDAP Follows Referral: Option specifying whether or not referrals to other LDAP servers are followed automatically.

LDAPv3 Repository Plugin Class Name: Where to find the class file that implements the LDAPv3 repository.

Attribute Name Mapping: Allows for common attributes known to the framework to be mapped to the native data store. Map the attributes as follows:

- `mail=userPrincipalName`
- `iplanet-am-user-alias-list=objectGUID`
- `employeeNumber=distinguishedName`
- `uid=sAMAccountName`
- `portalAddress=sAMAccountName`
- `telephonenumber=displayName`

LDAPv3 Plugin Supported Types and Operations: No change is needed.

LDAP Users Search Attribute: `cn` — Naming attribute of user.

LDAP Users Search Filter: `(objectclass=person)`

LDAP User Object Class: Object classes for user. When a user is created, this list of user object classes will be added to the user's attributes list. Therefore, it is important that the object classes you entered here actually exist in the Active Directory server; otherwise, you will get an object class violation (error=65).

Enter the following object classes (names are not case sensitive):

- `top`
- `person`
- `organizationalPerson`
- `user`

LDAP User Attributes: Definitive list of attributes associated with a user. If an attribute is not on this list, it will not be sent or read. Therefore, if there is any possibility that the user entry can contain this attribute, you should list it here. Or, if the attribute is not defined in the Active Directory server, you should not enter it here; otherwise, you will get an error when OpenSSO Enterprise tries to write this attribute to Active Directory. Enter the following attributes (names are not case sensitive):

- `cn`, `description`, `displayName`, `distinguishedName`, `dn`, `employeeNumber`, `givenName`, `mail`, `manager`, `memberOf`, `name`, `objectClass`, `objectGUID`, `postalAddress`, `sAMAccountName`, `sAMAccountType`, `sn`, `streetAddress`, `telephoneNumber`, `userAccountControl`, `userpassword`, `userPrincipalname`

- `iplanet-am-auth-configuration`, `iplanet-am-auth-login-success-url`, `iplanet-am-auth-login-failure-url`, `iplanet-am-auth-post-login-process-class`
- `iplanet-am-session-add-session-listener-on-all-sessions`, `iplanet-am-session-get-valid-sessions`, `iplanet-am-session-destroy-sessions`, `iplanet-am-session-max-caching-time`, `iplanet-am-session-max-idle-time`, `iplanet-am-session-max-session-time`, `iplanet-am-session-quota-limit`, `iplanet-am-session-service-status`
- `iplanet-am-user-auth-modules`, `iplanet-am-user-login-status`, `iplanet-am-user-admin-start-dn`, `iplanet-am-user-auth-config`, `iplanet-am-user-alias-list`, `iplanet-am-user-success-url`, `iplanet-am-user-failure-url`, `iplanet-am-user-password-reset-options`
- `iplanet-am-user-password-reset-question-answer`, `iplanet-am-user-password-reset-force-reset`, `sunIdentityServerDiscoEntries`, `iplanet-am-user-federation-info-key`, `iplanet-am-user-federation-info` `sunIdentityMSISDNNumber`
- `iplanet-am-user-admin-start-dn`, `iplanet-am-user-account-life`, `iplanet-am-user-alias-list`, `iplanet-am-user-auth-config`, `iplanet-am-user-failure-url`, `iplanet-am-user-login-status`, `iplanet-am-user-password-reset-force-reset`, `iplanet-am-user-password-reset-options`, `iplanet-am-user-password-reset-question-answer`, `iplanet-am-user-success-url`
- `sunAMAuthInvalidAttemptsData`
- `sunIdentityServerDeviceKeyValue`, `sunIdentityServerDeviceStatus`, `sunIdentityServerDeviceType`, `sunIdentityServerDeviceVersion`, `sunxmlkeyvalue`
- `sunIdentityServerPPFacadeNamePronounced`, `sunIdentityServerPPSignKey`, `sunIdentityServerPPDemographicsBirthday`, `sunIdentityServerPPCommonNameFN`, `sunIdentityServerPPDemographicsDisplayLanguage`, `sunIdentityServerPPCommonNameMN`, `sunIdentityServerPPLegalIdentityAltIDType`, `sunIdentityServerPPCommonNameAltCN`, `sunIdentityServerPPAddressCard`, `sunIdentityServerPPLegalIdentityAltIDValue`, `sunIdentityServerPPLegalIdentityMaritalStatus`, `sunIdentityServerPPLegalIdentityDOB`, `sunIdentityServerPPLegalIdentityVATIDValue`, `sunIdentityServerPPEncryptKey`, `sunIdentityServerPPMsgContact`, `sunIdentityServerPPDemographicsTimeZone`, `sunIdentityServerPPCommonNamePT`, `sunIdentityServerPPLegalIdentityGender`, `sunIdentityServerPPLegalIdentityVATIDType`, `sunIdentityServerPPDemographicsAge`, `sunIdentityServerPPFacadeGreetSound`, `sunIdentityServerPPEmploymentIdentityOrg`, `sunIdentityServerPPEmergencyContact`, `sunIdentityServerPPDemographicsLanguage`, `sunIdentityServerPPFacadeMugShot`, `sunIdentityServerPPFacadeGreetMeSound`, `sunIdentityServerPPFacadeWebSite`, `sunIdentityServerPPCommonNameCN`, `sunIdentityServerPPCommonNameSN`, `sunIdentityServerPPInformalName`,

sunIdentityServerPPEmploymentIdentityJobTitle, sunIdentityServerPPLegalIdentityLegalName, sunIdentityServerPPEmploymentIdentityAltO

User Status Attribute: `userAccountControl` — Attribute to check to determine if a user is active or inactive. When a user is created, the default user's active or inactive status is assigned based on the value in this field:

- User Status Active Value: 544
- User Status Inactive Value: 546

LDAP Groups Search Attribute: `cn` — Naming attribute of a group. This attribute name will be used to construct the group's `dn` and search filter.

LDAP Groups Search Filter: `(objectclass=group)` — Filter employed when doing a search for groups. The LDAP Groups Search Attribute will be prepended to this field to form the actual group search filter.

LDAP Groups Container Naming Attribute: `cn` — Naming attribute for a group container if groups resides in a container; otherwise, leave it blank.

LDAP Groups Container Value: `users` — Value for the group container.

LDAP Groups Object Class: `objectclasses` for group. When a group is created, this list of group object classes will be added to the group's attributes list. Enter the following object classes (names are not case sensitive):

- `group`
- `top`

LDAP Groups Attributes: Definitive list of attributes associated with a group. Any attempt to read or write group attributes that are not on this list is not allowed. Therefore, you should enter all possible attributes. Enter the following attributes (names are not case sensitive):

- `objectClass`
- `sAMAccountName`
- `distinguishedName`
- `member`
- `objectCategory`
- `dn`
- `cn`
- `sAMAccountType`
- `name`

Attribute Name for Group Membership: `memberOf` — Name of the attribute whose values are the names of all the groups that this `dn` belongs to.

Attribute Name of Unique Member: `member` — Attribute name whose value is a `dn` belonging to this group.

Attribute Name of Group Member URL: `memberUrl` — Name of the attribute whose value is an LDAP URL that resolves to members belonging to this group.

LDAP People Container Naming Attribute: `cn` — Naming attribute of people container if user resides in a people container.

LDAP People Container Value: `users`

LDAP Agents Search Attribute: `cn` — Naming attribute of an agent. This attribute name will be used to construct the agent's `dn` and search filter.

LDAP Agents Container Naming Attribute: `cn` — Naming attribute of agent container if agent resides in an agent container.

LDAP Agents Container Value: `users` — Value of the agent container.

LDAP Agents Search Filter: `(objectClass=sunIdentityServerDevice)` — Filter employed when searching for an agent.

LDAP Agents Object Class: `ojectclasses` for agents. When an agent is created, this list of user object classes will be added to the agent's attributes list. Enter the following object classes (names are not case sensitive):

- `person`
- `organizationalPerson`
- `sunIdentityServerDevice`
- `top`

LDAP Agents Attributes: Definitive list of attributes associated with a user. Any attempt to read or write user attributes that are not on this list is not allowed. Enter the following attributes (names are not case sensitive):

- `cn`
- `dn`
- `name`
- `objectClass`
- `userPassword`
- `sunIdentityServerDeviceVersion`
- `sunIdentityServerDeviceType`
- `sunIdentityServerDeviceKeyValue`
- `sunIdentityServerDeviceStatus`
- `sunxmlkeyvalue`
- `description`

Persistent Search Base DN: `DC=example,DC=com` — Base `DN` to use for a persistent search. For Active Directory, this needs to be the root suffix.

Persistent Search Maximum Idle Time Before Restart: Restart the persistence search if it has been idle for this maximum allowed time. Default value is OK.

Maximum Number of Retries After Error Codes: Number of times to retry the persistent search operation if it encounters the error codes specified in LDAP Exception Error Codes to Retry On. Default value is OK.

Delay Time Between Retries: Time to wait before each retry. Applies only to a persistent search connection. Default value is OK.

LDAP Exception Error Codes to Retry On: Retry the persistent search operations if these errors are encountered. Default value is OK.

5 **Click Finish.**

Configuring an Authentication Module to Login Through Active Directory

▼ To Configure an Authentication Module to Login Through Active Directory

1 **In the OpenSSO Administration Console, click realm for which you want to add the new authentication chain.**

2 **Click the Authentication tab.**

3 **Create a new module instance with the following data:**

- Primary Active Directory server: *ADServer*:*ADServerPort*
- DN to Start User Search: `dc=example,dc=com`
- DN for Root User Bind: `cn=Administrator,cn=users,dc=`*RootUser*`,dc=com`
- Password for Root User Bind: *AdministratorPassword*
- Attribute Used to Retrieve User Profile: *sAMAccountName*
- Attributes Used to Search for a User to be Authenticated: *sAMAccountName*
- Search Scope: `SUBTREE`

4 **Create a new Authentication chaining instance:**

a. **Add a new instance for the authentication instance created in the previous step.**

b. **Set the criteria to Sufficient.**

5 **Change Default Authentication Chain to the new authentication chain you just created.**

6 **Click Save.**

Next Steps To login using Active Directory for authentication, specify the following URL:

`http://`*YourAccessManagerServer*`:`*port*`/amserver/UI/login?org=`*YourRealmName*

Operational Notes

The above configuration will allow you to list users and groups. It will also allow you to perform some basic user profile operations. You should be able to change the following user profile information in the OpenSSO Console:

- `emailaddress`
- `employeeNumber`
- `telephonenumber` — Active Directory will add it.
- `postalAddress` — Home address in the Console. Active Directory will add it.
- `user` alias list

However, you cannot do the following operations because of missing attributes or object classes:

- Cannot create `firstname`, `lastname`, `fullname`.
- Cannot create a `group`.
- Cannot change the user authentication (`iplanet-am-user-auth-config`). No attribute exists.
- Cannot change the user status (`inetUserStatus`). No attribute exists.
- Cannot change the success URL (`iplanet-am-user-success-url`). No attribute exists.
- Cannot change the failure URL (`iplanet-am-user-failure-url`). No attribute exists.
- Cannot change the MSISDN number (`sunIdentityMSISDNNumber`). No attribute exists.
- Cannot create a user or agent in OpenSSO Console. The user must be created in Active Directory.
- Cannot change the user or agent password. This change must be done in Active Directory.

◆ ◆ ◆ CHAPTER 20

Using IBM Tivoli Directory Server as the User Data Store

This chapter describes how to configure IBM Tivoli Directory Server as the Sun OpenSSO Enterprise user data store.

Contents

Requirements For Using Tivoli Directory Server as the User Data Store

To configure and use Tivoli Directory Server as the user data store, your deployment must meet these requirements:

- OpenSSO Enterprise 8.0 (or an update or patch release) is installed configured on a supported web container.
- IBM Tivoli Directory Server 6.1 is running and accessible to OpenSSO Enterprise 8.0.

Loading LDIF Files for Tivoli Directory Server

Load the following LDIF files into IBM Tivoli Directory Server using your preferred directory server utility. These files contain LDAP attributes and object classes used by OpenSSO Enterprise for Tivoli Directory Server.

- `am_remote_tivolids_schema.ldif`
- `fam_tivolids_schema.ldif`

These files are available in the *zip-root*`/opensso/ldif` directory, where *zip-root* is where you unzipped the OpenSSO Enterprise distribution file.

Configuring the Tivoli Directory Server Data Store in the OpenSSO Console

▼ To Configure the Tivoli Directory Server Data Store in the OpenSSO Console

1 **Log in to the OpenSSO Administration Console.**

2 **Click Access Control,** *realm-name*, **Data Stores, and then New.**

3 **Enter the Name, check Generic LDAPv3, and then click Next.**

4 **On the New Data Store page, specify the following fields:**

LDAP Server: Fully qualified name and port number of the Tivoli Directory Server. For example: `tivolids.example.com:8080`

LDAP Bind DN: User DN who has sufficient access rights to Tivoli Directory Server. For exemple: `cn=root`

LDAP Bind Password: Password of the “LDAP Bind DN” user.

LDAP Organization DN: Base DN or starting point for this data store. For example: `dc=opensso,dc=java,dc=net`

LDAP SSL: Check to use an SSL connection.

LDAP Connection Pool Minimum Size: Use the default value 1.

LDAP Connection Pool Maximum Size: Use the default value 10.

Maximum Results Returned from Search: Use the default value 1000.

Search Timeout: Use the default value 10.

LDAP Follows Referral: Check `Enabled`.

LDAPv3 Repository Plug-in Class Name: Use the default value: `com.sun.identity.idm.plugins.ldapv3.LDAPv3Repo`

Attribute Name Mapping: Not required.

LDAPv3 Plug-in Supported Types and Operations: Operations that this data store can perform. Use the Current Values:

- `group=read,create,edit,delete`
- `realm=read,create,edit,delete,service`
- `user=read,create,edit,delete,servce`

LDAPv3 Plug-in Search Scope: Use the default value: `SCOPE_ONE`

LDAP Users Search Attribute: `cn`

LDAP Users Search Filter: `(objectclass=organizationalPerson)`

LDAP User Object Class: When a user is created, the user will be assigned these object classes. Depending on the object classes you have defined for your organization, some of the following default entries might not be necessary. If your organization has other object classes that are not on this list, add them to the list.

OpenSSO Enterprise requires these object classes: `iplanet-am-user-service`, `iplanetPreferences`, `sunFederationManagerDataStore`, `sunFMSAML2nameIdentifier`, and `sunIdentityServerLibertyPPService`.

`person`, `inetadmin`, `inetorgperson`, `inetUser` `iplanet-am-user-service`, `iplanetPreferences`, `organizationalperson`, `person` `sunFederationManagerDataStore`, `sunFMSAML2nameIdentifier`, `sunIdentityServerLibertyPPService`, `top`

LDAP User Attributes: List of attributes that can be assigned to a user. Depending on how you have configured your directory server, you might have to add or remove some of the entries in this list. OpenSSO Enterprise requires the attributes with the "`iplanet`" and "`sun`" prefixes.

`adminRole`, `authorityRevocationList`, `caCertificate`, `cn`, `distinguishedName`, `dn`, `employeeNumber`, `givenName`, `inetUserHttpURL`, `inetUserStatus`, `iplanet-am-auth-configuration`, `iplanet-am-user-auth-modules`, `iplanet-am-session-add-session-listener-on-all-sessions`, `iplanet-am-session-destroy-sessions`, `iplanet-am-session-get-valid-sessions`, `iplanet-am-session-max-caching-time`, `iplanet-am-session-max-idle-time`, `iplanet-am-session-max-session-time`, `iplanet-am-session-quota-limit`, `iplanet-am-session-service-status`, `iplanet-am-user-admin-start-dn`, `iplanet-am-user-account-life`, `iplanet-am-user-alias-list`, `iplanet-am-user-auth-config`, `iplanet-am-user-failure-url`, `iplanet-am-user-login-status`, `iplanet-am-user-password-reset-force-reset`, `iplanet-am-user-password-reset-options`, `iplanet-am-user-password-reset-question-answer`, `iplanet-am-user-success-url`, `iplanet-am-static-group-dn`, `mail`, `manager`, `memberOf`, `objectClass`, `postalAddress`, `preferredlanguage`, `preferredLocale`, `preferredtimezone`, `sn`, `sunAMAuthInvalidAttemptsData`, `sunIdentityMSISDNNumber`, `telephoneNumber`, `uid`, `userPassword`, `userCertificate`, `iplanet-am-user-federation-info-key`, `iplanet-am-user-federation-info`, `sunIdentityServerDiscoEntries`

`sunIdentityServerPPCommonNameCN`, `sunIdentityServerPPCommonNameFN`, `sunIdentityServerPPCommonNameSN`, `sunIdentityServerPPCommonNameMN`, `sunIdentityServerPPCommonNameAltCN`, `sunIdentityServerPPCommonNamePT`, `sunIdentityServerPPInformalName`, `sunIdentityServerPPLegalIdentityLegalName`, `sunIdentityServerPPLegalIdentityDOB`, `sunIdentityServerPPLegalIdentityMaritalStatus`, `sunIdentityServerPPLegalIdentityGender`, `sunIdentityServerPPLegalIdentityAltIdType`, `sunIdentityServerPPLegalIdentityAltIdValue`, `sunIdentityServerPPLegalIdentityVATIdType`, `sunIdentityServerPPLegalIdentityVATIdValue`, `sunIdentityServerPPEmploymentIdentityJobTitle`, `sunIdentityServerPPEmploymentIdentityOrg`, `sunIdentityServerPPEmploymentIdentityAltO`, `sunIdentityServerPPAddressCard`, `sunIdentityServerPPMsgContact`, `sunIdentityServerPPFacadeMugShot`, `sunIdentityServerPPFacadeWebSite`, `sunIdentityServerPPFacadeNamePronounced`, `sunIdentityServerPPFacadeGreetSound`, `sunIdentityServerPPFacadegreetmesound`, `sunIdentityServerPPDemographicsDisplayLanguage`, `sunIdentityServerPPDemographicsLanguage`, `sunIdentityServerPPDemographicsAge`, `sunIdentityServerPPDemographicsBirthDay` `sunIdentityServerPPDemographicsTimeZone` `sunIdentityServerPPSignKey`, `sunIdentityServerPPEncryPTKey`, `sunIdentityServerPPEmergencyContact`, `sun-fm-saml2-nameid-infokey`, `sun-fm-saml2-nameid-info`

Create User Attribute Mapping Current Values `cn` `sn`

Attribute Name of User Status: `inetuserStatus`

User Status Active Value: `Active`

User Status Inactive Value: `Inactive`

LDAP Groups Search Attribute: `cn`

LDAP Groups Search Filter: The filter to use when searching for a group. You might have change this value depending on which object class was used to denote a group: `(objectclass=groupOfNames)`

LDAP Groups container Naming Attribute:

LDAP Groups Container Value:

LDAP Groups Object Class Tivoli Directory Server 6.1 groups can be static, dynamic, and nested, but only a static group is supported by the Identity Repository (IdRepo) data store. A static group defines each member individually using the structural object class `groupofNames`, `groupOfUniqueNames`, `accessGroup`, or `accessRole`; or the auxilary object class `ibm-staticgroup` or `ibm-globalAdminGroup`. A static group using the structural object class `groupOfNames` and `groupOfUniqueNames` requires at least one member or `uniquemember`, respectively. `ibm-staticgroup` is the only class for which members is optional. All other object classes taking members require at least one member.

Only one type of group object class is supported by OpenSSO Enterprise. If you choose the type of group that requires at least one member, you must enter a user in "Default Group Member's User DN". This user will automatically be added to the group when a group is created. You can remove this user from the group after if you don't want this user to be a member of the group. accessGroup ibm-staticGroup top

LDAP Groups Attributes `ou` `dn` `objectclass` `cn` `uniqueMember` `description`

Attribute Name for Group Membership:

Attribute Name of Unique Member: `uniqueMember`

Attribute Name of Group Member URL: `memberUri`

Default Group Member's User DN: This user will be automatically added to the group when the group is created. This is necessary because when you create a group in the OpenSSO console, no users are assigned to the group. But most of the Tivoli Directory Server groups require at least one member when the group is created. For example: `cn=auser1,dc=opensso,dc=java,dc=net`

LDAP People Container Naming Attribute:

LDAP People Container Value:

Identity Types That Can Be Authenticated: Check `User`.

Authentication Naming Attribute: `uid`

Persistent Search Base DN: For example: `ou=company,dc=example,dc=com`

Persistent Search Filter: `(objectclass=*)`

Persistent Search Maximum Idle Time Before Restart: 0

The Delay Time Between Retries: 1000

Maximum Number of Retries After Error Codes: 3

LDAP Exception Error Codes to Retry On 80 81 91

Caching: Check `Enabled`.

Maximum Age of Cached Items: 600

Maximum Size of the Cache: 10240

5 **Click Finish.**

CHAPTER 21

Configuring OpenSSO Enterprise 8.0 in FIPS Mode

This chapter describes how to configure Sun OpenSSO Enterprise in Federal Information Processing Standards (FIPS) 140 mode, including:

This chapter described how to enable FIPS mode for Sun Java System Web Server 7.0. To enable FIPS mode for other web containers, refer to the product documentation for the specific web container.

Enabling FIPS Mode for the NSS Database

▼ To Enable FIPS Mode for the NSS Database

- **Enable FIPS mode for the NSS database using the Security Module Database Tool (`modutil`). For example:**

  `modutil -fips true -dbdir` *path-to-nss-database*

  where *path-to-nss-database* represents the path to the NSS database.

  For example, by default, for Web Server 7.0, the NSS database is in the `config` directory of the Web Server 7.0 instance.

  For information about using `modutil`, see `http://www.mozilla.org/projects/security/pki/nss/tools/modutil.html`.

Configuring FIPS Mode for Sun Java System Web Server 7.0

These procedures use Sun Java System Web Server 7.0 as the OpenSSO Enterprise web container with the NSS Certificate DB (`certdb`) as the key/certificate store.

- "Enabling FIPS Mode for Web Server 7.0" on page 168
- "Configuring the Web Server 7.0 Transport Layer Security (TLS) to be FIPS 140 Compliant" on page 169

Enabling FIPS Mode for Web Server 7.0

▼ To Enable FIPS Mode for Web Server 7.0

1. **If Web Server 7.0 has the Java Security Manager enabled, add the following additional permissions to the Web Server 7.0 `server.policy` file:**

```
permission java.security.SecurityPermission "insertProvider.Mozilla-JSS";
permission java.security.SecurityPermission "putProviderProperty.Mozilla-JSS";
permission java.security.SecurityPermission "removeProvider.Mozilla-JSS";
```

2. **Set the password for the internal PKCS11 token using either the Web Server 7.0 Administration Console or CLI command.**

For the password requirements in FIPS mode, see `http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140sp/140sp814.pdf`

For example, to set the password using the Web Server 7.0 `wadm` command:

```
wadm> set-token-pin --user=admin --password-file=admin.pwd
--host=serverhost --port=8989 --config=config1 --token=internal
```

Or, to set the password using the Web Server 7.0 Administration Console:

a. **In the Administration Console, go to the Configuration page.**

b. **Click Certificates and then PKCS11 Tokens.**

c. **Click the PKCS11 token name (default is internal).**

d. **Check the Token State box.**

e. **Enter the password information.**

f. **Click Save.**

3 **If you modified files in the Web Server 7.0 `config` directory using `modutil` or `certutil`, pull the changes into the Web Server 7.0 Admin Server. For example:**

```
wadm pull-config --user=admin --password-file=path-to-password-file
--host=server-host --port=8989 --config=config1 node1
```

4 **Confirm that FIPS is enabled by restarting the Web Server 7.0 instance. You should see a new prompt for the `certdb` password or PIN. For example:**

```
> Please enter the PIN for the "NSS FIPS 140-2 Certificate DB" token:
```

Configuring the Web Server 7.0 Transport Layer Security (TLS) to be FIPS 140 Compliant

▼ To Configure the Web Server 7.0 TLS to be FIPS 140 Compliant

1 **Log in to the Web Server 7.0 Administration Console.**

2 **Click Configuration.**

3 **Click the server instance you want to configure.**

4 **Click the HTTP Listeners tab and then click the listener instance you want to configure.**

5 **Select the SSL tab in new popup window.**

6 **Disable SSL2 and SSL3, leaving only TLS.**

7 **Disable all non-FIPS Compliant TLS Cipher suite by removing them from the Selected list.**

See the following list for the FIPS compliant TLS cipher suites.

8 **Save your changes.**

More Information FIPS Compliant TLS Cipher Suites

- TLS_DHE_DSS_WITH_AES_128_CBC_SHA
- TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
- TLS_DHE_DSS_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA

- TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA
- TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA
- TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA
- TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_ECDH_RSA_WITH_AES_128_CBC_SHA
- TLS_ECDH_RSA_WITH_AES_256_CBC_SHA

Configuring FIPS Mode for OpenSSO Enterprise 8.0

▼ To Configure FIPS Mode for OpenSSO Enterprise 8.0

Before You Begin

- `jss4.jar` file - The `jss4.jar` file must be compatible with the NSS version you are using. If necessary, download a compatible `jss4.jar` file and copy it to the application or web container `/lib` directory.
- Multiple OpenSSO Enterprise 8.0 instances - If you are configuring multiple OpenSSO Enterprise 8.0 instances that are part of a site, first add and configure all instances in the site in non-FIPS mode. Then, after all instances are added and configured for the site, configure the instances in FIPS mode.

1. **Log in to the OpenSSO Administration Console.**

2. **Click Configuration, Servers and Sites, and then the Server Name instance.**

3. **Click Security.**

4. **Click Inheritance Settings.**

5. **Uncheck the Encryption class, FIPS Mode, and Secure Random Factory Class properties.**

6. **Click Save and then Back to Server Profile.**

7. **Change Encryption class to** `com.iplanet.services.util.JSSEncryption`.

8. **Change Secure Random Factory Class to** `com.iplanet.am.util.JSSSecureRandomFactoryImpl`.

9 **Check Yes for FIPS Mode.**

10 **Click Save and then the Advanced tab.**

11 **Change the `com.iplanet.security.SSLSocketFactoryImpl` property to `com.iplanet.services.ldap.JSSSocketFactory`.**

12 **Click Add and add following property and value:**

- Property Name: `opensso.protocol.handler.pkgs`
- Property Value: `com.iplanet.services.comm`

13 **Click Add and add following property and value:**

- Property Name: `com.iplanet.am.admin.cli.certdb.dir`
- Property Value: *path-to-FIPS-enabled-NSS-database*

14 **Click Save.**

15 **Restart the OpenSSO Enterprise 8.0 server instance.**

OpenSSO Enterprise 8.0 FIPS Compliant Algorithms

OpenSSO Enterprise 8.0 uses the following FIPS compliant algorithms:

- Hashes: SHA-1
- Encryption: Triple Data Encryption Standard (Triple DES)
- XML signature (`XMLDSIG`):
  - `http://www.w3.org/2000/09/xmldsig#rsa-sha1`
  - `http://www.w3.org/2000/09/xmldsig#hmac-sha1`
  - `http://www.w3.org/2000/09/xmldsig#dsa-sha1`
- XML encryption:
  - Triple Data Encryption Standard (Triple DES)
  - AES_128_KeyWrap
  - AES_192_KeyWrap
  - AES_256_KeyWrap

◆ ◆ ◆ CHAPTER 22

Taking Precautions Against Session-Cookie Hijacking in an OpenSSO Enterprise Deployment

This chapter provides information about precautions you can take against specific security threats related to session-cookie hijacking in an OpenSSO Enterprise deployment.

A common concern for administrators who want to restrict access to web-based applications in an OpenSSO Enterprise deployment is that hackers might use applications, referred to as "rogue" or "untrusted," to hijack session cookies. This chapter describes the threat posed by this hacking technique and provides configuration steps you can perform to guard against this threat, as described in the following sections:

The tasks presented in this chapter apply to a deployment with the following components:

- Starting with OpenSSO Enterprise 8.0
- Starting with Policy Agent 3.0

Defining Key Cookie Hijacking Security Issues

The tasks presented in this chapter address specific security issues related to session-cookie hijacking. This section defines those security issues.

The term "cookie hijacking" simply refers to a situation where an impostor (a hacker, perhaps using an untrusted application) gains unauthorized access to cookies. Therefore, cookie hijacking, by itself, does not refer to unauthorized access to protected web resources. When the cookies being hijacked are session cookies, then cookie hijacking potentially increases the threat of unauthorized access to protected web resources, depending upon how the system is configured.

This section provides background information about specific security issues related to session-cookie hijacking, illustrating how this type of cookie hijacking is possible and how it can potentially lead to unauthorized access of protected web resources. This section provides the underlying basis for understanding how the tasks presented in this chapter enable OpenSSO EnterpriseOpenSSO Enterprise to handle the specified security issues.

OpenSSO Enterprise provides Single Sign-On (SSO) and Cross Domain Single Sign-On (CDSSO) features, enabling users to seamlessly authenticate across multiple web-based applications. OpenSSO Enterprise is able to provide this seamless authentication by using hypertext transfer protocol (HTTP) or secure hypertext transfer protocol (HTTPS) to set session cookies on users' browsers.

The way OpenSSO Enterprise is configured influences the way it sets the session cookies. Prior to the implementation of the tasks outlined in this chapter, OpenSSO Enterprise sets session cookies for the entire domain. Therefore, all applications hosted on the domain share the same session cookies. This scenario could enable an untrusted application to intervene and gain access to the session cookies. Specific configuration steps presented in this chapter address this issue. After you perform the configuration, OpenSSO Enterprise prevents different applications from sharing the same session cookies.

The following list provides some details about possible security issues related to session-cookie hijacking (labels, such as "Security Issue: Insecure Protocol," are used to make the issues easy to identify and discuss):

Security Issue: Insecure Protocol	An application does not use a secure protocol, such as HTTPS, which makes the session cookie prone to network eavesdropping.

The following security issues could apply if you do not perform the tasks presented in this chapter.

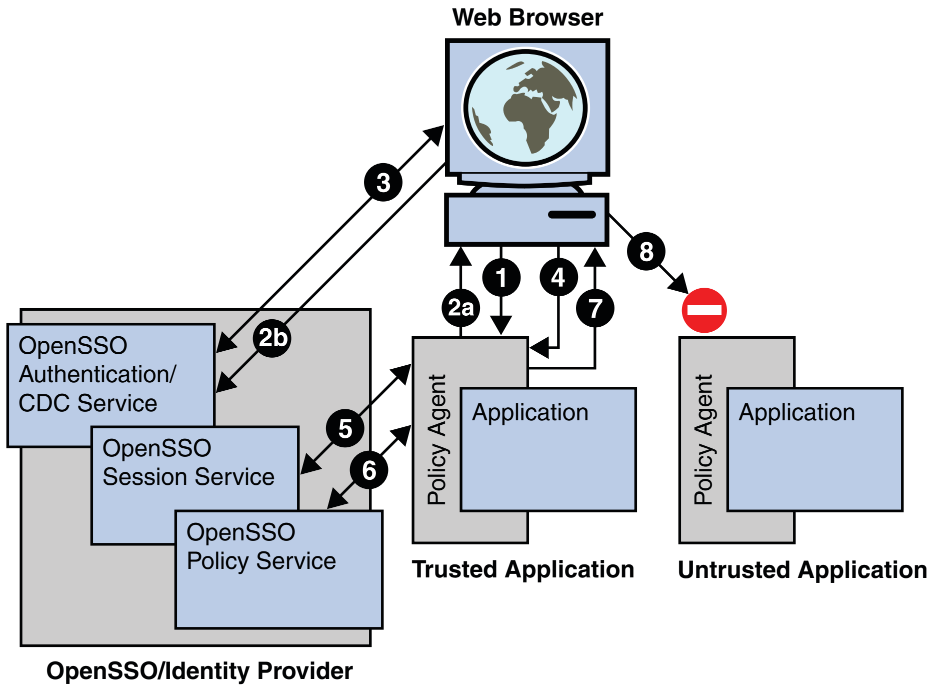
Security Issue: Shared Session Cookies	All applications share the same HTTP or HTTPS session cookie. This shared session-cookie scenario enables hackers to intervene by using an untrusted application to hijack the session cookie. With the hijacked session cookie, the untrusted application can impersonate the user and access protected web resources.
Security Issue: A Less Secure Application	If a single "less secure" application is hacked, the security of the entire infrastructure is compromised.
Security Issue: Access to User Profile Attributes	The untrusted application can use the session cookie to obtain and possibly modify the profile attributes of the user. If the user has administrative privileges, the application could do much more damage.

The figure illustrates a typical OpenSSO Enterprise deployment within an enterprise. While the figure helps to define security issues related to cookie hijacking, it also helps to define the solution. Therefore, the figure applies to a deployment where the tasks presented subsequently in this chapter have already been performed.

The deployment illustrated in the figure uses SSO. Moreover, as part of the solution, the OpenSSO Enterprise implementation of CDSSO is enabled. To guard against potential threats posed by cookie hijacking, CDSSO enablement is required regardless of the number of domains involved. Having CDSSO enabled when the deployment involves a single domain might seem counterintuitive, but ultimately security is increased by taking advantage of the CDSSO framework. For other information about the use of CDSSO in this type of deployment, see "Enabling OpenSSO Enterprise to Use Unique SSO Tokens" on page 179.

The numbers in the figure correspond to the numbered explanations that follow. The explanations combine to provide an outline of the process that takes place when a request is made for a protected resource, emphasizing how the SSO token flows between components. The deployment includes a single virtual AuthN (Authentication) and AuthZ (Policy) server (denoted as OpenSSO or Identity Provider in the figure), and a number of applications (Service Providers), denoted as Trusted Application and Untrusted Application in the diagram. The Service Providers are usually front-ended with an agent (specifically, an agent from the Policy Agent 3.0 software set) that handles the SSO (AuthN) and Policy (AuthZ).

FIGURE 22–1 Role of the Session Cookie in Allowing Access to Protected Web Resources



Note – The previous figure does not include every detail involved in the flow of the SSO token. The figure provides a simplified view that corresponds to the scope of this chapter.

---

1. The user accesses an application through a web browser. The agent (an agent from the Policy Agent 3.0 software set) intercepts the request and checks for the user's privilege to access the application. The check is specifically a check for a valid OpenSSO Enterprise SSO token, which is set as a session cookie in the user's browser.
2. Assuming the user does not have a valid SSO token, the agent redirects the browser to OpenSSO Enterprise Authentication Service. The agent also provides its identity using the Liberty AuthN request specification. Since this single redirection is a two step process, it is illustrated in the figure as two substeps: 2A and 2B.
3. OpenSSO Enterprise Authentication Service authenticates the user and, after successful authentication, redirects the user back to the target application with the SSO token as part of the URL query parameter (in a format specified by Liberty AuthN response specification).
4. The agent receives a successful AuthN response from OpenSSO Enterprise Authentication Service. It gets the SSO token and sets it as a session cookie for the host (rather than for the domain) to the browser's request.
5. The agent validates the SSO token with OpenSSO Enterprise Session Service.

6. After a successful SSO token validation in Step 5, the agent then checks the permissions for the user to access the application with OpenSSO Enterprise Policy Service.
7. If permission is granted in Step 6, the user is allowed access to the application.
8. As indicated in the figure with an incomplete connection to Untrusted Application, the same SSO token cannot be used to gain access to an untrusted application. OpenSSO Enterprise denies such access since the SSO token is unique to the application and may not be shared or reissued to other agents or applications.

Cookie Hijacking Security Issues

This section explains how performing the tasks described in this chapter enables OpenSSO Enterprise to handle the security issues discussed in the preceding section, “Defining Key Cookie Hijacking Security Issues” on page 173.

Note – When applications use a secure protocol such as HTTPS, the SSO Token is not visible to network snooping. This security issue is labeled “Security Issue: Insecure Protocol” in this chapter. Ensuring that all protected resources use a secure protocol is not a security measure administered using OpenSSO Enterprise, but this a very prudent security measure that you should consider implementing if it is not currently in place.

OpenSSO Enterprise Solution: Shared Session Cookies

The security issue labeled *Security Issue: Shared Session Cookies* in this chapter pertains to applications sharing the same HTTP or HTTPS session cookie. OpenSSO Enterprise addresses this security threat by issuing a unique SSO token to each Application/Agent after the user has been authenticated. The unique SSO token is referred to as a "restricted token."

The term “Application/Agent,” indicates that the restricted token is inextricably connected to the application and to the agent (which specifically refers to an agent from the Policy Agent 3.0 software set). Since each user's SSO token is unique for each Application/Agent, the increased security provided by this scenario prevents an non-trusted application, impersonating the user, from accessing other applications. More specifically, since the SSO token (restricted token) assigned to a user (as a part of the user's session) is associated with the agent that did the initial redirection for authentication, all subsequent requests are checked to verify that they are coming from the same agent. Thus, if a hacker tries to use the same restricted token to access another application, a security violation is thrown.

What makes the restricted token “restricted” is not related to the syntax of the token. The syntax of a restricted token is the same as that of a regular SSO token. Instead, a specific constraint is associated with the restricted token. This constraint is what ensures that the restricted token is only used for an application that a given agent protects.

OpenSSO Enterprise Solution: A Less Secure Application

The security issue labeled "Security Issue: A Less Secure Application" in this chapter pertains to the potential threat of applications that are "less secure." With the OpenSSO Enterprise solution, if one application is somehow compromised, the hacker cannot hack into other applications.

OpenSSO Enterprise Solution: Modification of Profile Attributes

The security issue labeled "Security Issue: Access to User Profile Attributes" in this chapter pertains to the threat posed by an untrusted application modifying the profile attributes of the user. The OpenSSO Enterprise solution to this issue does not change the SSO token. The restricted SSO token is similar to the regular SSO token ID. However, the set of Session Service operations that accept restricted SSO token IDs is limited. This functionality enables OpenSSO Enterprise to prevent applications from modifying profile attributes of the user.

Key Aspects of the OpenSSO Enterprise Solution: Cookie Hijacking Security Issues

The following subsections explain some of the key or more complex aspects of the OpenSSO Enterprise solution to the cookie hijacking security issues defined in this chapter.

OpenSSO Enterprise Session Cookies Involved in Issuing Unique SSO Tokens

When OpenSSO Enterprise is configured to issue unique SSO tokens for each Application/Agent, the following cookies are involved:

Cookie Name	Cookie Value (*place holder*)	Example Cookie Domain Information
iPlanetDirectoryPro	*SSO-token*	`OpenssoHost.example.com`

The value of this cookie, which is represented in the preceding table with the place holder *SSO-token*, is the actual value of the token. The domain is set to the host name of the OpenSSO Enterprise instance where the user was authenticated.

Cookie Name	Cookie Value (*place holder*)	Example Cookie Domain Information
iPlanetDirectoryPro	*restricted-SSO-token*	`agentHost.example.com`

The value of this cookie, which is represented in the preceding table with the place holder *restricted-SSO-token*, is the actual value of the token. The domain is set to the host name of the agent instance for which the restricted token is issued.

Cookie Name	Example Cookie Value	Example Cookie Domain Information
sunIdentityServerAuthNServer	`https://OpenssoHost.example.com:8080`	`.example.com`

The value of this cookie, which is represented in the preceding table with the example URL `https://OpenssoHost.example.com:8080`, is the URL of the OpenSSO Enterprise instance where the user was authenticated. The protocol used for this particular example is `HTTPS` while the port number is a non-default example, `8080`. The domain must be set such that it covers all the instances of OpenSSO Enterprise installed on the network.

Enabling OpenSSO Enterprise to Use Unique SSO Tokens

To enable OpenSSO Enterprise to issue unique SSO tokens, you must enable CDSSO. Therefore, though CDSSO is usually enabled for multiple-domain deployments, in this case, CDSSO must be enabled whether the entire deployment is on a single domain or is spread across multiple domains. In no way does enabling CDSSO for a single domain negatively affect the deployment.

The next section describes the steps required to configure OpenSSO Enterprise to prevent session-cookie hijacking from causing a breach of security.

Implementing the OpenSSO Enterprise Solution for Cookie Hijacking Security Issues

The instructions presented in this section provide a solution to the potential risks related to session-cookie hijacking as outlined in this chapter.

- "About the Agent Profile" on page 180

- "Configuring the OpenSSO Enterprise Deployment Against Cookie Hijacking" on page 180

This section also provides the other configuration steps necessary to guard web resources in an OpenSSO Enterprise deployment against the threat of session-cookie hijacking.

After you perform the tasks presented in this chapter, OpenSSO Enterprise starts enforcing restrictions on the sessions it creates. The new configuration enables OpenSSO Enterprise to more closely track aspects of each session. At that point, not only does OpenSSO Enterprise record which agent performed the initial redirection for authentication it also tracks the applications to which an SSO token has been issued. OpenSSO Enterprise uses this information to facilitate the processing of each subsequent request and to prevent unauthorized access to protected web resources.

About the Agent Profile

As a part of agent installation, each agent has its own agent profile. OpenSSO Enterprise server uses each agent's profile to help prevent cookie-hijacking related security issues. You can use the agent profile that was created as part of the initial agent installation, or if you choose, you can update the agent profile. If you choose to update the agent profile, this is the appropriate point to do so.

Configuring the OpenSSO Enterprise Deployment Against Cookie Hijacking

Though each agent has its own agent profile, OpenSSO Enterprise is not configured by default to associate an SSO token to a specific agent profile. The steps in this section enable this type of association. Ultimately, the new configuration introduces "restricted tokens" into the OpenSSO Enterprise deployment, guarding against security issues as described in this chapter.

▼ To Configure the OpenSSO Enterprise Deployment Against Cookie Hijacking

This task description includes configuration information for agents in the Policy Agent 3.0 software set. Perform the task on every agent instance for which you want to enhance security. The best practice is to perform the task on all the agent instances in the OpenSSO Enterprise deployment. As part of the configuration of each agent instance, you must also make specific configurations directly to OpenSSO Enterprise. For this task, be prepared to access the OpenSSO Enterprise Console and a browser that can access a protected web resource.

1. **Using a browser, navigate through OpenSSO Enterprise Console to the appropriate agent (J2EE agent or web agent, whichever applies) properties page that you want to configure.**

2 **Edit the agent properties as described in the substeps that follow:**

Notice, that you must navigate from Console tab to Console tab.

a. **Enable the property labeled Cross Domain SSO (Tab: SSO, Name:** `com.sun.identity.agents.config.cdsso.enable`**).**

Setting this property to Enabled, enables CDSSO, which is required for each agent instance since the agent will use functionality provided by the CDSSO feature.

b. **Set the property labeled CDSSO Servlet URL (Tab: SSO, Name:** `com.sun.identity.agents.config.cdsso.cdcservlet.url`**) as described in this substep.**

This property stores the URL to which you want to direct users after they log in successfully to a deployment enabled for CDSSO. A feasible setting for this property is as follows:

```
https://OpenssoHost.example.com:8080/amserver/cdcservlet
```

c. **Click Save on the SSO page.**

d. **(Conditional) For J2EE agents only, add a new value to the property labeled Custom Properties (Tab: Advanced, Name:** `com.sun.identity.agents.config.freeformproperties`**) as described in this step.**

Add the following value to the Custom Properties property:

```
com.sun.identity.enableUniqueSSOTokenCookie=true
```

e. **Click Save on the Advanced page.**

3 **Restart the container that hosts the agent.**

4 **Add the required OpenSSO Enterprise properties as described in the substeps that follow.**

a. **In the OpenSSO Enterprise Console, navigate back to the top level.**

b. **Click Configuration tab.**

c. **Click the Servers and Sites subtab.**

d. **Click the OpenSSO Enterprise server name that you esny to configure.**

e. **Click the Advanced tab.**

f. **Add the properties and values as shown in the table that follows.**

Property Name	Property Value
`com.sun.identity.enableUniqueSSOTokenCookie`	`true`

Property Name	Property Value
`com.sun.identity.authentication.uniqueCookieName`	`sunIdentityServerAuthNServer`
`com.sun.identity.authentication.uniqueCookieDomain`	*DomainName*. For example, `example.com`

g. Click Save.

5 In OpenSSO Enterprise Administration Console, navigate back to the Configuration tab.

6 Select the System subtab.

7 Click Platform.

8 In the Cookie Domain list, change the cookie domain name as described in the substeps that follow.

This step enables OpenSSO Enterprise to set host-specific session cookies instead of domain-wide session cookies.

a. Select the default domain, such as "example.com."

b. Click Remove.

c. Enter the name of the machine hosting the OpenSSO Enterprise instance.

For example:

```
OpenssoHost.example.com
```

d. Click Add.

9 Ensure that the proper cookies appear in a browser as described in the substeps that follow.

a. Use a browser to access a resource that is protected by the agent that you just configured.

b. Check the browser's cookie settings to ensure that the three following cookies appear:

Cookie Name	Example Cookie Value	Example Cookie Domain Information
iPlanetDirectoryPro	*SSO-token*	`OpenssoHost.example.com`
iPlanetDirectoryPro	*restricted-SSO-token*	`agentHost.example.com`
sunIdentityServerAuthNServer	`https://OpenssoHost.example.com:8080`	`.example.com`

CHAPTER 23

Patching OpenSSO Enterprise 8.0

Sun periodically provides update releases and patches to OpenSSO Enterprise 8.0 on `http://sunsolve.sun.com/`. To find the latest OpenSSO Enterprise 8.0 update release or patch, search for patch ID 141655.

An OpenSSO Enterprise 8.0 update release or patch includes a new `opensso.war` file that you can install using the methods described in "Planning Your Patch Operation" on page 183.

This chapter provides the following information about patching OpenSSO Enterprise:

- "Planning Your Patch Operation" on page 183
- "Overview of the `ssopatch` Utility" on page 185
- "Installing the `ssopatch` Utility" on page 187
- "Patching an OpenSSO Enterprise 8.0 WAR File" on page 188
- "Creating a New OpenSSO Enterprise 8.0 Patched WAR File" on page 190
- "Running the `updateschema` Script" on page 190
- "Patching a Site With Multiple OpenSSO Enterprise Instances" on page 191

Planning Your Patch Operation

This section describes the general steps to patch an OpenSSO Enterprise 8.0 release. If you are installing the `opensso.war` file in an update release or patch as a new deployment, see Chapter 3, "Installing OpenSSO Enterprise."

▼ To Plan Your Patch Operation

1. **Check the patch and upgrade paths in Table 23–1.**

2. **If you are not familiar with `ssopatch`, read the "Overview of the `ssopatch` Utility" on page 185.**

3. **Before you install an OpenSSO Enterprise 8.0 update release or patch, check the information about the new features, hardware and software requirements, and issues and workarounds in the following documents:**
   - *Sun OpenSSO Enterprise 8.0 Release Notes*
   - *Sun OpenSSO Enterprise 8.0 Update 1 Release Notes*

4. **Download the lastest OpenSSO Enterprise update release or patch (patch ID 141655) from** `http://sunsolve.sun.com/`.

5. **Check the** `README` **file associated with the patch for information such as the issues fixed in the patch.**

6. **Install** `ssopatch` **for your platform, as described in "Installing the** `ssopatch` **Utility" on page 187.**

7. **Determine if your existing WAR file has been customized or modified, by comparing your WAR file to its internal manifest file.**

8. **Compare your existing WAR file and the update release WAR file, to return the files customized in the original WAR, files updated in the new WAR file, and files added or deleted between the two WAR versions.**

9. **Backup your existing OpenSSO WAR file and configuration data.**

10. **Patch your OpenSSO Enterprise WAR file, as described in "Patching an OpenSSO Enterprise 8.0 WAR File" on page 188.**

11. **Run the** `updateschema` **script, as described in "Running the** `updateschema` **Script" on page 190.**

OpenSSO Patch and Upgrade Paths

TABLE 23–1 OpenSSO Patch and Upgrade Paths

Release	Patch or Upgrade Path
Sun OpenSSO Enterprise 8.0 update release or patch	■ Patching OpenSSO Enterprise 8.0 with an update release or patch using the `ssopatch` utility. ■ Patching the following OpenSSO Enterprise 8.0 specialized WAR files with an update release using the `ssopatch` utility: ■ OpenSSO Administration console only WAR file ■ Distributed Authentication UI server WAR file ■ OpenSSO server only WAR file, without the Administration Console ■ IDP Discovery Service WAR file ■ Patching an update release or patch with a newer update release or patch using the `ssopatch` utility. ■ Installing an OpenSSO Enterprise 8.0 update release or patch as a new deployment. ■ Creating and installing a new specialized WAR file from an OpenSSO Enterprise 8.0 update release or patch.
Sun Java System Access Manager 7.1 Sun Java System Access Manager 7 2005Q4 Sun Java System Federation Manager 7.0	To go from Access Manager 7.x or Federation Manager 7.0 to an OpenSSO Enterprise 8.0 update or patch release: 1. Upgrade to OpenSSO Enterprise 8.0, as described in the *Sun OpenSSO Enterprise 8.0 Upgrade Guide*. 2. Apply the update release or patch, as described in this chapter.

Overview of the `ssopatch` Utility

The `ssopatch` utility is a Java command-line utility that is available on Solaris and Linux systems as `ssopatch` and on Windows systems as `ssopatch.bat`.

Note – The syntax for `ssopatch` in OpenSSO Enterprise 8.0 update releases has changed considerably since the OpenSSO Enterprise 8.0 release. For the new syntax, see Running the `ssopatch` Utility in this section.

The `ssopatch` patch utility can perform these functions:

- Compares an OpenSSO Enterprise WAR to its original manifest, to determine if the WAR file has been customized or modified
- Compare two OpenSSO Enterprise WAR files, to determine the differences between the two files including any customizations made to the original WAR file and any changes in the new WAR file
- Generates a staging area of the files required to generate a new patched OpenSSO Enterprise WAR file

The `ssopatch` utility uses a manifest file to determine the contents of a specific OpenSSO Enterprise WAR file. A manifest file is an ASCII text file that contains:

- A string that identifies the specific version of the OpenSSO Enterprise WAR file
- All of the individual files in the OpenSSO Enterprise WAR file, with checksum information for each file

The manifest file is usually named `OpenSSO.manifest` and is stored in the in the `META-INF` directory of the OpenSSO Enterprise WAR file. The `ssopatch` utility sends its results to the standard output (`stdout`). If you prefer, you can capture the `ssopatch` output by redirecting the output to a file. If `ssopatch` finishes successfully, it returns a zero (0) exit code. If errors occur, `ssopatch` returns a non-zero exit code.

Running the `ssopatch` Utility

To run the `ssopatch` utility, follow this usage:

```
ssopatch --help|-? [--locale|-l]

ssopatch --war-file|-o [--manifest|-m] [--locale|-l]

ssopatch --war-file|-o
--war-file-compare|-c
[--staging|-s]
[--locale|-l]
[--override|-r]
[--overwrite|-w]
```

where the options are:

Option	Description
`--war-file|-o`	Specifies a path to a WAR file (such as `opensso.war`) that has previously been deployed.

Option	Description
`--manifest|-m`	Specifies the path to the manifest file you want to create. The manifest file will be generated from the WAR file indicated by `--war-file|-o`, if this option is provided.
`--war-file-compare|-c`	Specifies a path to a WAR file to compare against the WAR file indicated by `--war-file|-o`.
`--staging|-s`	Specifies a path to the staging area where the files from an OpenSSO Enterprise WAR will be written.
`--locale|-l`	Specifies the locale to be used. If this option is not specified, `ssopatch` uses the default system locale.
`--override|-r`	Overrides revision checking for the two WAR files. Revision checking determines the versions of the WAR files and continues only if the versions are compatible. This option allows you to override this check. The fefault is false (revision checking is performed).
`--overwrite|-w`	Overwrites the files in the existing staging area. The default is false (files are not overwritten).

Installing the `ssopatch` Utility

Before you install the `ssopatch` utility:

- Download and unzip the patch ZIP file in the latest OpenSSO Enterprise 8.0 update or patch release (patch ID 141655).
- Set your `JAVA_HOME` environment variable point to JDK 1.5 or later.

Note – Always install the latest version of the `ssopatch` or `ssopatch.bat` utility (and the corresponding `updateschema.sh` or `updateschema.bat` script) from the OpenSSO Enterprise 8.0 patch release you are installing.

▼ To Install the `ssopatch` Utility

1 **Locate the `ssoPatchTools.zip` file in the *zip-root*`/opensso/tools` directory, where *zip-root* is where you unzipped the update release patch ZIP file.**

2 **Create a new directory to unzip the `ssoPatchTools.zip` file. For example:** `ssopatch-files`

3 **Unzip the `ssoPatchTools.zip` file in the new directory. You then get these files:**

- `README`

- `ssopatch` and `ssopatch.bat` utilities
- `resources` directory, which contains the `ssopatch` properties files
- `lib` directory, which contains the `ssopatch` JAR files
- `patch` directory, which contains the `updateschema` and `updateschema.bat` scripts and related XML files

4 **If you want to run the `ssopatch` utility from a directory other than its current directory without providing the full path, add the utility to your `PATH` variable.**

Patching an OpenSSO Enterprise 8.0 WAR File

The patching operation compares the manifests for each WAR file and then shows:

- Files customized in the original OpenSSO Enterprise 8.0 WAR file
- Files updated in a new OpenSSO Enterprise 8.0 update release WAR file
- Files added or removed between the two WAR file versions

The `ssopatch` utility then copies the appropriate files to a staging directory, where you must add any customizations before you create and deploy the new patched WAR file.

▼ To Patch OpenSSO Enterprise 8.0

Before You Begin Although `ssopatch` does not modify your original `opensso.war` file, it is recommended that you back up this file, in case you need to back out the patched `opensso.war` file. Backup your existing OpenSSO Enterprise WAR file and configuration data:

- Copy your existing OpenSSO Enterprise WAR file to a safe location. Then, if you need to back out an update for some reason, you can re-deploy the backup copy of your WAR file.
- Backup your configuration data, as described in the *Sun OpenSSO Enterprise 8.0 Administration Guide*.

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

2 **Although `ssopatch` does not modify your original `opensso.war` file, it is recommended that you back up this file, in case you need to back out the patched `opensso.war` file.**

3 **Run `ssopatch` to create the staging area. For example:**

```
./ssopatch -o /zip-root/opensso/deployable-war/opensso.war
-c /u1/opensso/deployable-war/opensso.war --override -s /tmp/staging

Generating Manifest for: /zip-root/opensso/deployable-war/opensso.war
Original manifest: Enterprise 8.0 Build 6(200810311055)
New manifest: Enterprise 8.0 Update 1 Build 6.1(200904300525)
```

```
Versions are compatible
Generating Manifest for: /u1/opensso/deployable-war/opensso.war
Comparing manifest of /zip-root/opensso/deployable-war/opensso.war (generated-200905051031)
  against /u1/opensso/deployable-war/opensso.war (generated-200905051032)
File was customized in original, but not found in new war.
Staging area using original war version (samples/saml2/sae/header.jsp)
File was customized in original, but not found in new war.
Staging area using original war version (WEB-INF/template/opends/config/upgrade/config.ldif.4517)
File was customized in original, but not found in new war.
Staging area using original war version (WEB-INF/template/opends/config/upgrade/schema.ldif.4517)
Differences: 1813
Customizations: 0
```

In this example, `/tmp/staging` is the staging area where `ssopatch` copies the files.

4 Update the files as needed in the staging area, using the results of the previous step.

The following table shows the potential results of the patch operation and the actions you might need to take.

`ssopatch` **Results**	**Explanation and Action Required**
`File not in original war (`*filename*`)`	The indicated file does not exist in the original OpenSSO WAR but is in the latest version of the OpenSSO WAR. **Action**: None
`File updated in new war (`*filename*`)`	The indicated file exists in both the original and new OpenSSO WAR files and has been updated in the latest version of the OpenSSO WAR. No customizations have been done in the original OpenSSO WAR. **Action**: None
`File customized (`*filename*`)`	The indicated file exists in both OpenSSO WAR files, has been customized in the original version of the WAR, but has not been updated in the latest version of the WAR. **Action**: None
`May require manual customization (`*filename*`)`	The file exists in both OpenSSO WAR files, has been customized in the original version of the WAR, and has been updated in the latest version of the WAR. **Action**: if you want your customizations in the file, you must manually add them to the new updated file in the staging directory.
`File was customized in original, but not found in new war`	The file existed in the original WAR file, but is not in the new WAR. **Action**: None.

Creating a New OpenSSO Enterprise 8.0 Patched WAR File

▼ To Create a New OpenSSO Enterprise 8.0 Patched WAR File

1. **Create a new OpenSSO Enterprise WAR file from the files in the staging area. For example:**

   ```
   cd /tmp/staging
   jar cvf /patched/opensso.war *
   ```

   where `/patched/opensso.war` is the patched OpenSSO Enterprise WAR 8.0 file.

2. **Redeploy the `/patched/opensso.war` file to the web container using the original deploy URI. For example: `/opensso`**

Running the `updateschema` Script

After you run `ssopatch`, run the `updateschema.sh` on Solaris or Linux systems or `updateschema.bat` on Windows. The script updates the OpenSSO Enterprise server version, adds new default server properties, adds new attribute schemas required for fixes and enhancements in the OpenSSO Enterprise 8.0 update release. You must run `updateschema` in order to update the server version.

▼ To Run the `updateschema` Script

Before You Begin

- The `updateschema.sh` or `updateschema.bat` script requires the OpenSSO Enterprise 8.0 Update 1 or later version of the `ssoadm` command-line utility. Therefore, before you run this script, install the latest admin tools from the OpenSSO Enterprise 8.0 update release tools, as described in "Installing the OpenSSO Enterprise Utilities and Scripts in the `ssoAdminTools.zip` File" on page 76.
- Also, always run the latest version of the `updateschema.sh` or `updateschema.bat` script from the patch release you are installing.

1. **Change to the *patch-tools*`/patch` directory, where *patch-tools* is where you unzipped `ssoPatchTools.zip`.**

2. **Run `updateschema.sh` or `updateschema.bat`.**

3. **When the scripts prompts you, provide the following information:**
   - Full path to the `ssoadm` utility (excluding `ssoadm` itself). For example: `/opt/ssotools/opensso/bin`

- `amadmin` password

The `updateschema.sh` or `updateschema.bat` script writes any messages or errors to the standard output.

Next Steps **OpenSSO Enterprise 8.0 Update Release Configuration Changes**. The patched OpenSSO Enterprise 8.0 WAR file might have configuration changes that were not in your original WAR file. Any configuration changes, if any, will be documented separately for the patch. Check the patch documentation and the Release Notes for more information about any configuration changes. (The version string in the OpenSSO manifest file will change, even if there are no configuration changes in the new WAR file.)

Patching a Site With Multiple OpenSSO Enterprise Instances

In this scenario, you have configured OpenSSO Enterprise as a site with multiple server instances and you want to patch each server instance.

▼ To Patch a Site With Multiple OpenSSO Enterprise Instances

1. **Patch the OpenSSO Enterprise WAR file on each OpenSSO Enterprise server instance, as described in "Patching an OpenSSO Enterprise 8.0 WAR File" on page 188.**

2. **On the first OpenSSO Enterprise instance only, run the `updateschema` or `updateschema.bat` script, as described in "Running the `updateschema` Script" on page 190.**

3. **Restart the web container for each OpenSSO Enterprise server instance.**

CHAPTER 24

Uninstalling OpenSSO Enterprise

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. **Stop the OpenSSO Enterprise web container.**

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

- *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_enterprise_80.zip` and extracted files.**

Troubleshooting **OpenSSO data store port**. If the OpenSSO Enterprise server instance was using the OpenSSO data store, the data store port was in use by the LISTEN socket. Stopping the web container server instance or domain should release this port. To check the data store port, use the `netstat` command. For example, if the OpenSSO data store used default port 50389:

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

Port 50389 should not be in use for the LISTEN socket. If necessary, release this port.

Uninstalling the OpenSSO Enterprise Utilities and Scripts

▼ To Uninstall the OpenSSO Enterprise Utilities and Scripts

1 **Remove the directory and its contents where `ssoAdminTools.zip` was extracted.**

2 **Optionally, remove the `ssoAdminTools.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 **Stop the Distributed Authentication UI server web container.**

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 **Stop the web container.**

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 **Stop the web container.**

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 **Stop the web container.**

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 **Stop the web container.**

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_s1ds_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_s1ds_schema.ldif`
- `fam_sds_schema.ldif`

Index