Oracle® OpenSSO 8.0 Update 2 Release Notes

Part No: 821–1857
February 2012

ORACLE

120207@25097

Contents

Preface

The Oracle OpenSSO 8.0 Update 2 Release Notes provide information about downloading and installing OpenSSO Update 2 software including patch releases. This document also contains information about changes to the software since the OpenSSO Update 1 release.

Who Should Use This Book

These Release Notes are meant to be used by enterprise administrators and developers who are deploying Oracle OpenSSO 8.0 Update 2 including patch releases. You should already be familiar with concepts and procedures described in the core product documentation.

Related Books

These Release Notes supplement the core Oracle OpenSSO 8.0 product documentation at the following URL:

`http://docs.oracle.com/cd/E19681-01/index.html`

Related Third-Party Web Site References

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

Note – Oracle is not responsible for the availability of third-party web sites mentioned in this document. Oracle 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. Oracle 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://www.oracle.com/technetwork/indexes/documentation/index.html`)
- Support (`http://www.oracle.com/us/support/systems/index.html`)
- Training (`http://www.oracle.com/us/education/selectcountry-new-079003.html`) – Choose the country for which you want Training information for former Sun products.

Oracle Software Resources

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/technetwork/tutorials/index.html`).
- Download Sample Code (`http://www.oracle.com/technetwork/indexes/samplecode/index.html`).

Typographic Conventions

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

TABLE P–1 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.`

TABLE P–1 Typographic Conventions *(Continued)*

Typeface	Meaning	Example
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–2 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#`

Revision History

Part Number	Date	Description
821–1857–14	February 2012	Revised Chapter 2, "OpenSSO 8.0 Update 2 Patch Releases," for patch 4.
821–1857–13	June 2011	Revised Chapter 2, "OpenSSO 8.0 Update 2 Patch Releases," for patch 3.

Part Number	Date	Description
821–1857–12	March 2011	■ Revised Chapter 2, "OpenSSO 8.0 Update 2 Patch Releases," for patch 2. ■ Revised outdated URLs.
821–1857–11	December 2010	Added Chapter 2, "OpenSSO 8.0 Update 2 Patch Releases."
821–1857–10	July 2010	Initial release.

◆ ◆ ◆ CHAPTER 1

About OpenSSO 8.0 Update 2

This chapter contains the following topics:

What's New in OpenSSO 8.0 Update 2

OpenSSO 8.0 Update 2 includes enhancements to the Security Token Service and the OpenSSO Fedlet. This update also includes new web container support for WebLogic 10.3.3 and fixes to many bugs.

Security Token Service Enhancements

The Security Token Service now includes the following new features:

- Supports TokenType for generating a specific web service provider security token.
- Supports both Asymmetric and Transport binding for X509 and username security tokens as requestor.
- Enforces SSL/Transport binding with a username security token when OpenSSO STS is configured with a username over SSL.
- Issues SAML holder-of-key security token for Asymmetric KeyType with useKey as the web service client public key and web service client X509 security token.
- WSDL is dynamically updated based on security token configuration.
- Supports encryption by the web service provider public key.
- Encrypts the static username password before storing it in the configuration store.

- Supports UserName token as On Behalf Of security token through a WS-Trust request.
- Supports issuance of SAML Bearer Tokens.
- New Web Service Security authentication module WSSAuth supports digest password validation.
- New OAMAuth authentication module enables single sign-on using Oracle Access Manager with OpenSSO.

For more information, see Chapter 4, "Using the Security Token Service."

Fedlet Enhancements

The Fedlet now includes the following new features:

- Supports encryption in the .NET Fedlet
- Supports signing in the .NET Fedlet
- .NET Fedlet now supports single logout
- .NET Fedlet provides Service Provider initiated single sign-on and artifact support
- Supports multiple Identity Providers and Identity Provider Discovery in .NET Fedlet
- Supplies version information within property and configuration files for the Fedlet
- New password SPI implementation
- Supports attribute query
- Supports single logout

For more information, see Chapter 5, "Using the Oracle OpenSSO Fedlet."

Bugs Fixed in This Release

The table lists issues that have been resolved in OpenSSO 8.0 Update 2.

TABLE 1–1 Bugs Fixed in This Release

Change Request Identifier	Description
6422249	SAML assertions using excessive memory.
6659356	New bug with the interaction process in a load-balanced scenario.
6802207	Policy agent "gateway servelet" function yields "Your authentication module is denied."
6894077	In Cookie hijacking mode, logout request hangs.
6931544	Javadoc comments missing for public API `AMLoginModule:isSessionQuotaReached.`

TABLE 1–1 Bugs Fixed in This Release *(Continued)*

Change Request Identifier	Description
6918266	`/opensso/realm/IDRepoEdit` delete Session service configuration in realm.
6923660	Inheritance setting in agent profile does not work as expected.
6924534	`ssoadm` --version did not return the right value after patching 141655-03.
6926203	`goto` URL not validated on distributed authentication.
6928480, 6934888	Distributed authentication UI: In log files IP recorded is DAUI IP, not client IP.
6931012	Access Manager console becomes unresponsive after adding a new config property.
6931476	Incorrect exceptions thrown in the logs for misconfigured SAML/IDP's service URLs on the Service Provider side.
6933168	Password reset page is not localized when locale parameter is given in the URL.
6933268	"Auth module instance" condition with "application timeout properties" set drops session after login.
6937698	OpenSSO8.0: Console Invalid Characters check is not performed
6937700	OpenSSO allows to create username with special characters, but complains during login.
6939038	Security Token Service client samples are failing for IBM Websphere Application Server 6.1.
6940455	Security Token Service "ssoadm set-site-sec-urls" throws an NPE on the console.
6942485, 6942813	OpenSSO does not escape "\" in `uid` correctly, and 2 different `uid` values are stored in Directory Server entry.
6945286	Distributed Authentication login: `uid` with special characters results in error.
6947033	“URL not found” exception errors in SAML.
6949778	`iplanet-am-auth-locale` value of realm is not taken in consideration in the evaluation process.
6947068	`goto` is missing after session timeout.
6958448	`LDAPv3Repo.setAttributes` method fetches the schema multiple times even for a single modification.

Hardware and Software Requirements For OpenSSO 8.0 Update 2

See the *System Requirements and Supported Platforms for Oracle OpenSSO 8.0u2* document listed under Oracle Branded Releases of Sun Products Supported Configuration at the following URL:

`http://www.oracle.com/technetwork/middleware/ias/downloads/fusion-certification-100350.html`

OpenSSO 8.0 Update 2 Issues and Workarounds

General Security Advisory

General security concerns exist regarding using a HTTP Basic Authentication module. See `http://en.wikipedia.org/wiki/Basic_access_authentication`, the "Disadvantages"

section. Be sure that you can address these security concerns before you consider using HTTP Basic Authentication in a production deployment.

CR 6959610: OpenSSO 8.0 Update 2 samples should be removed in production environment

To minimize random or unnecessary configuration changes through inadvertent sample program runs, remove the samples before you deploy OpenSSO 8.0 Update 2 in a production environment.

CRs 6944573, 6964648: New Java security permissions are required for WebLogic Server 10.3.3

If you are deploying OpenSSO 8.0 Update 2 on Oracle WebLogic Server 10.3.3 with the security manager enabled, an additional Java security permission is required.

Workaround. Add the following permission to the WebLogic Server 10.3.3 `weblogic.policy` file:

```
permission java.lang.RuntimePermission "getClassLoader";
```

CR 6939443: Certificate authentication with LDAP checking or OCSP checking fails on WebLogic Server 10.3.x

Due to an issue in earlier versions of Oracle WebLogic Server such as 10.3.0 and 10.3.1, certificate authentication with either LDAP checking or OSCP checking enabled fails.

Workaround. This problem has been fixed in WebLogic Server 10.3.3. To use certificate authentication with either LDAP checking or OSCP checking, use OpenSSO Update 2 with WebLogic Server 10.3.3.

CR 6960514: Cannot access authentication certificates

In the Spanish version of OpenSSO 8.0 Update 2, you cannot access authentication certificates. When you go to Configuration > Authentication > Certificates, an error occurs. The following is displayed in the log "Caused by: java.lang.IllegalArgumentException."

Workaround. None.

▼ To Configure JDBC Authentication with Oracle Database

1 **Download the `ojdbc6.jar` file from the following URL:**

`http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-112010-090769.html`.

2 **Create a staging area and change to that directory. For example:**

```
mkdir /tmp/staging
cd /tmp/staging
```

3 **Explode the `opensso.war` in the staging area.**

```
jar xf opensso.war
```

4 **Change to the `WEB-INF/lib` directory.**

5 **Copy `ojdbc6.jar` into that directory. For example:**

`cp` *OJDBC6_DOWNLOAD_LOCATION*`/ojdbc6.jar`

6 **Create an updated `opensso.war` file from the staging area. For example:**

```
cd ../..
jar cf /tmp/opensso.war *
```

7 **Undeploy the current `opensso.war`.**

8 **Deploy the `opensso.war` file you created in Step 6.**

9 **Restart the OpenSSO web container instance.**

▼ To Manually Configure NSS on OpenSSO

By default, the OpenSSO configurator supports only the JCE/JSSE provider for SSL. However, you can use the OpenSSO administration console to manually enable JSS/NSS. If OpenSSO is deployed on Sun Web Server 7.0 or on GlassFish Enterprise Edition 2.1.0, then complete the following steps. For GlassFish Enterprise Edition 2.1.1 and later versions, see "CR 6967026: Configurator cannot connect to LDAPS-enabled directory server" on page 17.

Before You Begin

- If you want OpenSSO to connect to an LDAPS-enabled directory server, then the CA certificate for the LDAPS-enabled directory server must be already imported into the JVM trust store (by default *JAVA_HOME*`/jre/lib/security/cacert`).

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

2. **Click Configuration > Servers and Sites > Server Name instance.**
3. **Click Security.**
4. **Click Inheritance Settings.**
5. **Uncheck the Encryption class and Secure Random Factory Class properties.**
6. **Click Save, and then click 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. **Click Save, and then click the Advanced tab.**
10. **Change the `com.iplanet.security.SSLSocketFactoryImpl` property to `com.iplanet.services.ldap.JSSSocketFactory`.**
11. **Edit the following property and value:**
    - Property Name: `opensso.protocol.handler.pkgs`
    - Property Value: `com.iplanet.services.comm`
12. **Click Add, and add following property and value:**
    - Property Name: `com.iplanet.am.admin.cli.certdb.dir`
    - Property Value: *path-to-NSS-database*
13. **Click Save.**
14. **Restart the OpenSSO Enterprise 8.0 server instance.**

CR 6967026: Configurator cannot connect to LDAPS-enabled directory server

If OpenSSO is deployed on GlassFish Enterprise Server 2.1.1 or later versions, then OpenSSO cannot connect to an LDAPS-enabled directory server instance with JSS/NSS. The problem occurs because OpenSSO and GlassFish Enterprise Server 2.1.1 and later versions do not use the same JSS version.

Workaround: Use the JSSE provider instead of the NSS provider for SSL.

CR 6948937: Activating OpenSSO 8.0 Update 2 in WebLogic Server 10.3.3 admin console causes exceptions

If you deploy OpenSSO 8.0 Update 2 (`opensso.war`) in the WebLogic Server 10.3.3 administration console and click Start to allow OpenSSO 8.0 Update 2 to start receiving requests, exceptions are thrown in the console where the WebLogic Server domain was started.

Note: After you start OpenSSO 8.0 Update 2, it remains started and exceptions are not thrown again until OpenSSO 8.0 Update 2 is stopped and then restarted.

Workaround. Copy the `saaj-impl.jar` file from the OpenSSO 8 Update 2 `opensso-client-jdk15.war` file to the WebLogic Server 10.3.3 configuration `endorsed` directory, as follows:

1. Stop the Oracle WebLogic Server 10.3.3 domain.
2. If necessary, unzip the OpenSSO 8.0 Update 2 `opensso.zip` file.
3. Create a temporary directory and unzip the *zip-root*`/opensso/samples/opensso-client.zip` file in that directory, where *zip-root* is where you unzipped the `opensso.zip` file. For example:

   ```
   cd zip-root/opensso/samples
   mkdir ziptmp
   cd ziptmp
   unzip ../opensso-client.zip
   ```

4. Create a temporary directory and extract the `saaj-impl.jar` file from `opensso-client-jdk15.war`. For example:

   ```
   cd zip-root/opensso/samples/ziptmp/war
   mkdir wartmp
   cd wartmp
   jar xvf ../opensso-client-jdk15.war WEB-INF/lib/saaj-impl.jar
   ```

5. Create a new directory named `endorsed` under the `WEBLOGIC_JAVA_HOME/jre/lib` directory (if `endorsed` does not already exist), where `WEBLOGIC_JAVA_HOME` is the JDK that WebLogic Server is configured to use.
6. Copy the `saaj-impl.jar` file to the `WEBLOGIC_JAVA_HOME/jre/lib/endorsed` directory.
7. Start the WebLogic Server domain.

CR 6956461:SecurID authentication fails on IBM WebSphere Application Server

When OpenSSO is configured on IBM WebSphere Application Server 6.1 or AIX 5.3, a valid plain text password user can not be authenticated via a SecurID authentication module instance.

Workaround. None. Do not use plain text passwords on IBM WebSphere Application Server.

CR 6959373: Web container requires a restart after running `updateschema` script

After you run the `updateschema.sh` or `updateschema.bat` script, you must restart the OpenSSO 8.0 Update 2 web container.

CR 6961419: Running `updateschema.bat` script requires a password file

The `updateschema.bat` script executes several `ssoadm` commands. Therefore, before you run `updateschema.bat` on Windows systems, create a password file that contains the password user in clear text for the `amadmin` user. The `updateschema.bat` script prompts you for the path to the password file. Before the script terminates, it removes the password file.

CR 6970859: Browser scroll feature does not work

When using OpenSSO Update 2 on the following browsers, the browser scroll does not work as designed: Microsoft Internet Explorer 7 and 8 on Windows 2003 or 2008.

Workaround. Maximize the browser window.

Deploying OpenSSO 8.0 Update 2 on JBoss 5.0

JBoss 5.x uses Tomcat 6.0.16 which does not support the special symbols in the OpenSSO iPlanetDirectoryPro cookie. This affects OpenSSO cookie-handling.

Workaround. See "To Deploy OpenSSO on JBoss 5.0" on page 19.

▼ To Deploy OpenSSO on JBoss 5.0

Before You Begin

- The minimum heap size should be set to at least 512M (-Xms256m), and maximum heap size should be set to 1024M (-Xmx1024m).
- The `MaxPermSize` should be set to 256M (-XX:MaxPermSize=256m)

1 In the JBoss `run.conf` file (`run.conf.bat` on Windows), which is used to start up the JBoss instance, add the following JVM options:

```
-Dcom.iplanet.am.cookie.encode=true
-Dcom.iplanet.am.cookie.c66Encode=true
```

If you do not set these properties, after entering your credentials in the OpenSSO console, you are directed back to the login page. After you've deployed and configured OpenSSO you can remove this entry in the `run.conf` file (or `run.conf.bat` on Windows). OpenSSO configures the cookie encode property during deployment.

2 Unjar the opensso.war.

a. Create text-file `opensso.war/WEB-INF/jboss-web.xml`.

b. Enter the following content in the file:

```
<!DOCTYPE jboss-web PUBLIC "-//JBoss//DTD Web Application 5.0//EN"
"http://www.jboss.org/j2ee/dtd/jboss-web_5_0.dtd">
<jboss-web>
<class-loading java2ClassLoadingCompliance='true'>
        <loader-repository>
            jbia.loader:loader=opensso
            <loader-repository-config>
                java2ParentDelegaton=true
            </loader-repository-config>
        </loader-repository>
   </class-loading>
   <resource-ref>
        <res-ref-name>jdbc/openssousersdb</res-ref-name>
        <jndi-name>java:jdbc/openssousersdb</jndi-name>
    </resource-ref>
</jboss-web>
```

3 Create the WAR again.

4 Stop the JBoss server.

5 Create a directory under the mode that opensso will be deployed to.

Example: `JBOSS_INSTALL_DIR>/server/`*$CONFIG*`/deploy/opensso.war`

where *$CONFIG* is the mode such as default, all, or production.

6 Go to the `opensso.war` directory.

Example: *JBOSS_INSTALL_DIR*`/server/`*$CONFIG*`/deploy/opensso.war`

7 Explode the war to this directory.

`jar -xvf` *WAR_FILE_LOCATION*`/opensso.war`

8 Restart the JBoss container.

Deployment of `opensso.war` will succeed without errors.

Note – OpenSSO 8.0 U2 installation on JBoss 5.0.0 is supported in exploded war mode only.

CR 6971437 : OpenSSO 8.0 Update 2 loses configuration after restart of JBoss Application Server 5.0.0.0

If you deploy and configure the `opensso.war` file on JBoss Application Server 5.0.0.0 and then restart the JBoss Application Server web container, OpenSSO 8.0 Update 2 displays the configurator page again instead of the login page.

Workaround. Deploy the `opensso.war` file in the JBoss AS deploy directory, as follows:

1. Stop the JBoss Application Server web container.
2. Edit the JBoss Application Server `run.conf` file by adding the following options:

```
-Dcom.iplanet.am.cookie.encode=true
-Dcom.iplanet.am.cookie.c66Encode=true
```

3. Uncomment the line "`admin=admin`" in the following files:
   - `JBOSS_INSTALL_DIR/server/$CONFIG/conf/props/jmx-console-users.properties`
   - `JBOSS_INSTALL_DIR/server/$CONFIG/deploy/management/console-mgr.sar/web-console.war/WEB-INF/classes/web-console-users.properties`
4. Copy the `opensso.war` file to the following JBoss Application Server directory:

   `JBOSS_INSTALL_DIR/`*server/$CONFIG*`/deploy`

   where *$CONFIG* is the JBoss Application Server mode, such as default, all, or production.
5. Restart the JBoss Application Server web container.
6. Deploy the `opensso.war` file in the directory shown in Step 4.

CR 6972593: Java Oracle OpenSSO Fedlet single sign-on (SSO) fails on JBoss AS 5.0.x

If you deploy the Java Oracle OpenSSO Fedlet on JBoss Application Server 5.0.x, index.jsp doesn't display and Fedlet SSO fails with an `IllegalStateException`.

Workaround. Follow these steps.

1. Stop the JBoss AS web container. JBoss AS web container.
2. Add the following Java options in the JBoss AS 5.0 run.conf file: -

```
Djavax.xml.soap.MetaFactory=
com.sun.xml.messaging.saaj.soap.SAAJMetaFactoryImpl
-Djavax.xml.soap.MessageFactory=
com.sun.xml.messaging.saaj.soap.ver1_1.SOAPMessageFactory1_1Impl
-Djavax.xml.soap.SOAPConnectionFactory=
com.sun.xml.messaging.saaj.client.p2p.HttpSOAPConnectionFactory
```

```
-Djavax.xml.soap.SOAPFactory=
com.sun.xml.messaging.saaj.soap.ver1_1.SOAPFactory1_1Impl
```

3. Start the JBoss AS web container.

SR 72335286 and CR 6929674: LDAP Referrals Do Not Work as Expected

When LDAP referrals are enabled, authentication fails for the user in the referral directory server. Authentication fails regardless of how the option "LDAP Follows Referral" is set. Also, the Subjects tab in the OpenSSO administration console does not display referral users.

These issues are due in part because of a known issue with the LDAP SDK (CR 6969674). Using LDAP SDK, LDAP referrals are not honored in OpenSSO.

Workaround. There are no workarounds at this time.

OpenSSO 8.0 Update 2 Documentation

In addition to this document, additional OpenSSO 8.0 documentation is available in the following collection:

`http://download.oracle.com/docs/cd/E19681-01/index.html`

Documentation Issues

OpenSSO 8.0 Update 2 includes the following documentation issues:

CR 6958580: Console online Help documents unsupported Discovery Agents

The OpenSSO 8.0 Update 2 administration console online Help documents Discovery Agents, even though these agents are not supported.

Workaround. None. Ignore the information about Discovery Agents in the online Help.

CR 6967006 Console online Help does not document OAMAuth and WSSAuth authentication modules

The OpenSSO 8.0 Update 2 administration console online Help does not document the Oracle Access Manager (OAM) and Web Services Security (WSS) authentication modules.

Workaround. For information about these authentication modules, see Chapter 4, "Using the Security Token Service."

CR 6953582: Fedlet Java API reference should be public

The Fedlet Java API public reference is available as part of the Oracle OpenSSO 8.0 Update 2 Java API Reference, which is available in the following documentation collection:

`http://download.oracle.com/docs/cd/E19681-01/index.html`

Note: OpenSSO 8.0 Update 2 does not support the `getPolicyDecisionForFedlet` method, even though this method is in the Java API reference.

CR 6953579: OpenSSO Fedlet README file should document single logout feature

The Fedlet README files do not document the single logout feature.

Workaround. For Oracle OpenSSO 8.0 Update 2, the Fedlet single logout feature is documented in Chapter 5, "Using the Oracle OpenSSO Fedlet."

CR 6960630: Information for patching a specialized OpenSSO WAR should be revised

The information has been revised. See "Patching a Specialized OpenSSO WAR" on page 48.

Additional Information and Resources

You can also find additional useful information and resources at the following locations:

- Oracle Advanced Customer Services for Systems:

  `http://www.oracle.com/us/support/systems/advanced-customer-services/index.html`

- Software Products: `http://www.oracle.com/us/sun/sun-products-map-075562.html`

- My Oracle Support: https://support.oracle.com/
- Oracle Technology Network: http://www.oracle.com/technetwork/index.html
- Sun Developer Services: https://shop.oracle.com/pls/ostore/f?p=ostore:2:0::NO:RP,2:PROD_HIER_ID:14755487300180585563861

Deprecation Notifications and Announcements

- The Service Management Service (SMS) APIs (`com.sun.identity.sm` package) and SMS model are no longer included in OpenSSO.
- The Unix authentication module and the Unix authentication helper (`amunixd`) will not be included in a future OpenSSO release.
- The Sun Java System Access Manager 7.1 Release Notes stated that the Access Manager `com.iplanet.am.sdk` package, commonly known as the Access Manager SDK (AMSDK), and all related APIs and XML templates will not be included in a future OpenSSO release.

  Consequently, when the AMSDK is removed, the Legacy Mode option and support will also be removed.

  Migration options are not available now and are not expected to be available in the future. Oracle Identity Manager provides user provisioning solutions that you can use instead of the AMSDK. For more information about Identity Manager, see http://www.oracle.com/us/products/middleware/identity-management/index.html.

How to Report Problems and Provide Feedback

If you have questions or issues with OpenSSO 8.0 Update 2 or a subsequent patch release, contact support resources at https://support.oracle.com/.

This site has links to the Knowledge Base, Online Support Center, and Product Tracker, as well as to maintenance programs and support contact numbers. If you are requesting help for a problem, please include the following information:

- Description of the problem, including when the problem occurs and its impact on your operation
- Machine type, operating system version, web container and version, JDK version, and OpenSSO version, including any patches or other software that might be affecting the problem
- Steps to reproduce the problem
- Any error logs or core dumps

Accessibility Features for People With Disabilities

To obtain accessibility features that have been released since the publishing of this media, consult Section 508 product assessments available upon request to determine which versions are best suited for deploying accessible solutions.

For information about Oracle's commitment to accessibility, see `http://www.oracle.com/index.html`.

Related Third-Party Web Sites

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

Note – Oracle is not responsible for the availability of third-party Web sites mentioned in this document. Oracle 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. Oracle will not be responsible or liable for any actual or alleged damage or loss caused by or in connection with the use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

◆ ◆ ◆ CHAPTER 2

OpenSSO 8.0 Update 2 Patch Releases

Oracle periodically releases patches for OpenSSO 8.0 Update 2. This chapter provides the following information about these patch releases:

About OpenSSO 8.0 Update 2 Patch Releases

The following information applies to all OpenSSO 8.0 Update 2 patch releases:

- Patches are available on the My Oracle Support site: `https://support.oracle.com/`. After you sign in, search for the patch ID under the Patches & Updates tab.
- For information about installing a patch, see Chapter 3, "Installing OpenSSO 8.0 Update 2."
- Patches are cumulative. You can install the latest patch without first installing an earlier patch. However, if you did not install an earlier patch, review the earlier patch sections to determine if any of the features or issues apply to your deployment.
- For a list of the problems fixed in a patch, see the `README` file distributed with the patch.

OpenSSO 8.0 Update 2 Patch 4

OpenSSO 8.0 Update 2 patch 4 is available as patch ID **141655-08** on the My Oracle Support site. Information about this patch includes:

- "Bug 13361224: SecurID authentication support for WebSphere Application Server 6.1 on AIX 6.1" on page 29

Bug 12286933: Dist Auth cannot receive session notifications

In patch 4, the new `com.sun.identity.client.notification.url` property in the `AMDistAuthConfig.properties` file allows a Distributed Authentication UI (DAUI) deployment to receive session notifications. This property replaces the `com.iplanet.am.notification.url` property.

For a DAUI deployment, the `com.sun.identity.client.notification.url` property defines the URL where notifications will be received by the client application, in the following format:

protocol`://`*host*`:`*port*`/`*distauth-uri*`/notificationservice`

For a new DAUI deployment, no changes are required, because the new property is available by default in the `AMDistAuthConfig.properties` file. However, in the case of a DAUI deployment upgrade from an older version, you must reconfigure the DAUI deployment after upgrading and redeploying the Dist Auth WAR file, because the original `AMDistAuthConfig.properties` does not have this property.

Otherwise, if you do not reconfigure the DAUI deployment, this property must be manually added to the `DistAuthConfig.properties` file of the upgraded instance.

Redeploying the Dist Auth WAR file is required, but if you reconfigure, you do not have to add the property manually. If you don't reconfigure the DAUI deployment, you must manually add the property after redeploying.

Bug 12427762: SAML attributes containing a | are not decoded in a SAML attribute

In patch 4, the new `com.sun.identity.saml.escapespecialchars` property determines if the special characters "|" and "&" should be escaped during attribute mapping in a generated session after SAML SSO by a Service Provider.

By default `com.sun.identity.saml.escapespecialchars` is set to `true`, which specifies that the characters should be escaped.

If you do not want the special characters to be escaped (that is, you want the characters retained as they are now), set the property to `false`, as follows:

In the Oracle OpenSSO Admin Console, click Configuration > Servers and Sites > Server SP > Advanced > and then set the `com.sun.identity.saml.escapespecialchars` property to `false`.

Bug 13361224: SecurID authentication support for WebSphere Application Server 6.1 on AIX 6.1

For SecurID authentication to operate with IBM WebSphere Application Server 6.1 on the AIX 6.1 platform, the SecurID Java Authentication APIs must be updated. You must replace the existing SecurID Java Authentication API JAR files in the OpenSSO WAR file (`opensso.war`) with the latest RSA Authentication API for Java version 8.1.1.312.

Download the SecurID Java Authentication API JAR files from the RSA website:

`http://www.rsa.com/`

These JAR files must replaced in the `opensso.war` file:

- `authapi.jar`
- `cryptoj.jar`
- `log4j-1.2.8.jar`

To replace the JAR files in the `opensso.war`:

1. Create a staging directory.
2. Explode the `opensso.war` in the staging directory.
3. Copy the new SecurID JAR files to the *staging-directory*`/opensso/WEB-INF/lib` directory.
4. Recreate the `opensso.war` file from the staging directory.
5. Deploy the `opensso.war`.

   Note: If the `opensso.war` is already deployed, first undeploy the existing `opensso.war` and then redeploy the updated `opensso.war`.
6. Restart the OpenSSO web container.
7. Configure the SecurID authentication module as described in the Oracle OpenSSO documentation in the following library:

   `http://docs.oracle.com/cd/E19681-01/index.html`
8. Restart the OpenSSO web container

OpenSSO 8.0 Update 2 Patch 3

OpenSSO 8.0 Update 2 patch 3 is available as patch ID **141655-07** on the My Oracle Support site. Other information about this patch includes:

- "Known Issues in OpenSSO 8.0 Update 2 Patch 3" on page 30
- "Documentation Updates in OpenSSO 8.0 Update 2 Patch 3" on page 30

Known Issues in OpenSSO 8.0 Update 2 Patch 3

Bug 12308272: OpenSSO `list-agents` command fails with GlassFish v2.1.1 patch 9

Other issues related to this bug include:

- Bug 12361318: OpenSSO 8.0 Update 2 patch 1 `ssoadm` command returns null pointer exception with GlassFish v2.1.1 patch 10
- Bug 12305906: Convergence SSO is not working when OpenSSO is deployed with GlassFish v2.1.1 patch 7 and later

These problems occur with GlassFish v2.1.1 patch 7 and later patches because of an incompatibility with the JAX-RPC JAR files.

Workaround. Downgrade to GlassFish v2.1.1 patch 6.

Documentation Updates in OpenSSO 8.0 Update 2 Patch 3

Bug 12307986: OpenSSO client SDK caches URL policy decision with correct methods

In patch 3, the OpenSSO client SDK caches the URL policy decision with all correct methods for a policy. Previously, only the URL policy decision for the method being accessed was cached.

For the Policy Service to return the policy actions for a given policy, the following property must be set in the OpenSSO client SDK configuration:

```
com.sun.identity.policy.client.cache.combine.actionItems.enabled=true
```

By default, this value is set to false.

Bug 12309423: Inconsistent session timeout behavior is fixed

Patch 3 fixes an inconsistent session timeout behavior. In some cases, OpenSSO server displayed the Login page rather than the Session Timeout page.

However, for the Session Timeout page to be displayed, the Purge Delay value must be greater than 0 (zero).

OpenSSO 8.0 Update 2 Patch 2

OpenSSO 8.0 Update 2 patch 2 is available as patch ID **141655-06** on the My Oracle Support site. Other information about this patch includes:

What's New in OpenSSO 8.0 Update 2 Patch 2

CR 7016248: Validation of `gotoOnFail` URLs

OpenSSO 8.0 Update 2 Patch 2 can validate a `gotoOnFail` URL after a user fails authentication. This validation prevents a hacker from sending the user to an imposter site.

To set valid `gotoOnFail` URLs, follow these steps after you install patch 2:

1. If you patched an earlier version of OpenSSO 8.0, make sure you have run the `updateschmema.sh` or `updateschema.bat` script and then restarted the OpenSSO web container, as described in "Running the `updateschema` Script" on page 49.
2. In the OpenSSO Administration Console, click Access Control, *realm-name*, Authentication, and then Advanced Properties.
3. Under Valid gotoOnFail URL domains, add each valid goto domain name, as follows:
   - A domain name starting with a dot (.) such as `.example.com` allows all hosts in the `example.com` domain to be used in a failure redirect URL.
   - A domain name that does not start with a dot (.) such as `example.com` allows the host `example.com` to be used in a failure redirect URL.

     For example, `http://example.com` would be valid, but `http://host.example.com` would not be valid.
   - If you don't add the entire domain to the list, you must add each individual agent host name being used.
   - You do not need to add domains for agents in CDSSO mode, because they are protected automatically.
4. Click Save.
5. Log out of the console and restart the OpenSSO web container.

Additional Information

- If a `gotoOnFail` URL is found to be invalid, the user is redirected to the default login failure URL.
- If you subsequently want to disable the `gotoOnFail` URL validation, remove all entries from the Valid goto URL domains list.

CR 6993122: SAMLv2 implementation of `NameIDPolicy` interface without `SPNameQualifier`

OpenSSO 8.0 Update 2 Patch 2 provides an implementation of the `NameIDPolicy` interface without the `SPNameQualifier` attribute.

The `SPNameQualifier` attribute in the `NameIDPolicy` interface is optional in a SAMLv2 authentication request. In some instances, a service provider (SP) initiated SSO can fail because an identity provider (IDP) cannot recognize the `SPNameQualifier` attribute in `NameIDPolicy` of the authentication request.

This implementation is available in the following new class:

```
com.sun.identity.saml2.protocol.impl.NameIDPolicyImplWithoutSPNameQualifier
```

The default behavior (that is, to put the `SPNameQualifier` attribute in `NameIDPolicy` of the authentication request) does not change.

To use the new class, follow these steps:

1. In the OpenSSO Administration Console, click Configuration, Servers and Sites, *server-name*, and then Advanced.
2. Add the following new property and value:
   - Property: `com.sun.identity.saml2.sdk.mapping.NameIDPolicy`
   - Value: `com.sun.identity.saml2.protocol.impl.NameIDPolicyImplWithoutSPNameQualifier`
3. Click Save.
4. Logo out of the console and restart the OpenSSO server web container.

`HttpServletRequest` and `HttpServletResponse` are available with Distributed Authentication User Interface (6677966)

OpenSSO 8.0 Update 2 Patch 2 allows you to access the `HttpServletRequest` object and modify the `HttpServletResponse` object through a custom authentication module for OpenSSO server deployments with the Distributed Authentication User Interface (DAUI), as well as for OpenSSO server deployments without the DAUI.

To use this new feature, you must modify your existing custom authentication modules using the authentication SPI framework. (If you don't want to use this feature, your existing custom authentication modules do not need to be modified. The current APIs for `getHttpServletRequest` and `getHttpServletResponse` will continue to be supported but only for OpenSSO server deployments without the DAUI.)

Changes to custom authentication modules include both JAVA class files and callback XML files. No UI changes are required. OpenSSO 8.0 Update 2 Patch 2 adds these new callbacks:

- `HttpRequestCallback`: equivalent to the container `HttpServletRequest` object
- `HttpResponseCallback`: equivalent to the container `HttpServletResponse` object

For more information, see the *OpenSSO Enterprise 8.0 Developer's Guide*.

Known Issues in OpenSSO 8.0 Update 2 Patch 2

CR 7017520: Missing property in Policy Service causes HTTP status code 500

For OpenSSO 8.0 Update 2 Patch 1 and later releases, the Policy Service sometimes returns HTTP status code 500. This problem is caused by a missing `app_sso_token_invalid` key in the `amPolicy.properties` file.

Workaround:

1. In the *OpenSSO-Deploy-base*`/WEB-INF/classes/amPolicy.properties` file, add the following line:

   ```
   app_sso_token_invalid=Application sso token is invalid
   ```

   OpenSSO-Deploy-base represents the path where the web container deploys the `opensso.war` file.
2. Restart the OpenSSO web container.

Documentation Updates in OpenSSO 8.0 Update 2 Patch 2

- "CR 7013849: Documentation update: WS-Trust certificate must be the same on client and server" on page 34
- "CR 7007193: Documentation update: REST Get method parameter passing is changed in OpenSSO 8.0 Update 2" on page 34

CR 7013849: Documentation update: WS-Trust certificate must be the same on client and server

The *Oracle OpenSSO STS Administrator's Guide* requires additional information about the Private Key Alias in Chapter 4, Managing the Security Token Service:

`http://download.oracle.com/docs/cd/E17842_01/doc.1111/e17844/tokenservice.htm`

Private Key Alias

Behind the Private Key Alias, a real certificate exists in the client's keystore. The value of this certificate depends on the OpenSSO server configuration. For authentication between a web services client (WSC) and a web services provider (WSP) such as OpenSSO server to function properly, the certificates on the client and OpenSSO server must match.

On the client side, you must import the certificate from OpenSSO server into the client's certificate store database. This imported certificate can be under a different name than OpenSSO server, but the client and OpenSSO server must use the same certificate to communicate properly.

For more information about web services security, see the *OpenSSO Enterprise 8.0 Administration Reference*:

`http://download.oracle.com/docs/cd/E19681-01/820-3886/index.html`

CR 7007193: Documentation update: REST Get method parameter passing is changed in OpenSSO 8.0 Update 2

OpenSSO 8.0 Update 2 and later releases do not allow sensitive information such as a password in URLs using the REST identity interface. This change (CR 6940612) prevents sensitive information from appearing in browser history files and web server or proxy log files.

If you are using the REST identity interface, a URL that contains sensitive information such as a password returns an unsupported operation exception. For example, the follow URL contains the user's password and would return an exception:

`https://opensso.example.com:80/opensso/identity/authenticate?username=`*user*`&password=`*user-password*

In the *OpenSSO Enterprise 8.0 Developer's Guide*, Chapter 10, Using the REST Identity Interfaces, states that "the REST authenticate interface works with simple user name and password only." However, in OpenSSO 8.0 Update 2 and later releases, sensitive information such as the password is not allowed in the URL and returns an exception.

Therefore, if you are using the REST identity interface with OpenSSO 8.0 Update 2 and later releases, use a POST operation to send the authentication data to OpenSSO server. POST data is usually not logged or stored as part of the browser history.

OpenSSO 8.0 Update 2 Patch 1

OpenSSO 8.0 Update 2 patch 2 is available as patch ID **141655-05** on the My Oracle Support site.

Known Issues in OpenSSO 8.0 Update 2 Patch 1

CR 6978018: Running OpenSSO 8.0 in GlassFish 2.1.x using LDAPS with JDK 1.6.x

To run OpenSSO 8.0 in a GlassFish 2.1.x web container with an external directory server using LDAPS with JDK 1.6.x, set the `NSS_USE_DECODED_CKA_EC_POINT` environment variable to 1 before you start the GlassFish 2.1.x domain. For example:

```
NSS_USE_DECODED_CKA_EC_POINT=1
export NSS_USE_DECODED_CKA_EC_POINT
glassfish-root/bin/asadmin start-domain glassfish-domain
```

CR 7002787: OpenSSO 8.0 Update 2 is not working with Active Directory Data Store

This problem occurs for both OpenSSO 8.0 Update 2 and OpenSSO 8.0 Update 2 patch 1. If you create an Active Directory data store and then log in to the OpenSSO administration console using the Active Directory authentication module, OpenSSO returns the error message "User has no profile in this organization" to your browser.

Workaround. To use the Active Directory data store and authentication module with OpenSSO 8.0 Update 2 or OpenSSO 8.0 Update 2 patch 1, perform these steps:

1. Log in to the OpenSSO Administration Console.
2. Under the Active Directory data store configuration, make these changes:

a. For the LDAPv3 Plug-in Supported Types and Operations, change:

```
user=read,create,edit,delete
```

to

```
user=read,create,edit,delete,service
```

b. In Attribute Name Mapping, add the following attribute mappings:

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

c. Click Save and log out of the console.

3. Restart the OpenSSO web container.

CR 6897101: After a login to a non-default realm, user experiences multiple logins after a timeout

Previously, if a user entered valid credentials after an authentication module timeout occurred, the login screen for the second authentication module was presented and the user could enter an invalid password to get access to a protected resource.

Patch 1 fixes this CR; however, this fix works only with non-JAAS modules. If you write a custom authentication module, you must use non-JAAS modules.

CR 6983035: Remote console with OpenSSO server returns errors after a session timeout

If you log in to OpenSSO server from a remote console and a session timeout occurs, some console functions do not work properly. Also, errors are displayed if you click on various tabs in the console.

Workaround. After making changes from the remote console, log out from the remote console. To get rid of the errors, restart both OpenSSO server and the remote console.

CR 6983026: Remote console with OpenSSO server causes errors when modifying Federation or SAML v2 attributes requiring the certificate keystore

If you are using a remote console and try to save Federation or SAML properties that need access to the certificate keystore, errors are returned. This problem occurs because the certificate keystore resides on the OpenSSO server, and the remote console does not have access to the keystore.

Workaround. Use either of these solutions, depending on your deployment:

- If the keystore is directly accessible from the remote console through a mount point, specify the complete absolute path to the keystore.
- Copy the keystore files from the OpenSSO server to the remote console. This solution, however, requires that if you make changes to the keystore files on the OpenSSO server, you must also update the keystore files on the remote console.

CR 6995584: "Post-Authentication Plug-In for First Time Login" sample requires OpenSSO 8.0 Update 1 or later

If you are using the sample in "Example 1–1 Code Sample: Post-Authentication Plug-In for First-Time Login" in the *Sun OpenSSO Enterprise 8.0 Integration Guide*, you must be running OpenSSO 8.0 Update 1 or later. Otherwise, the sample does not compile because the Java compiler cannot find the `POST_PROCESS_LOGIN_SUCCESS_URL` property, which was first available with OpenSSO 8.0 Update 1.

◆ ◆ ◆ CHAPTER 3

Installing OpenSSO 8.0 Update 2

This chapter contains the following topics:

OpenSSO 8.0 Update 2 Installation Overview

OpenSSO 8.0 Update 2 is available as a patch at the following URL:

`http://www.oracle.com/technetwork/middleware/downloads/oid-11g-161194.html`

Before you install OpenSSO 8.0 Update 2 (or subsequent patches), check the information about new features, hardware and software requirements, and issues and workarounds in this document.

OpenSSO 8.0 Update 2 includes an `opensso.war` file that you can install using these methods:

- **Patch an existing OpenSSO 8.0 deployment**: Use the `ssopatch` utility in Update 2 to patch an existing OpenSSO 8.0 deployment, as described in this chapter.

  Note - Oracle supports patching only OpenSSO 8.0 releases. For example, patching OpenSSO 8.0 with OpenSSO 8.0 Update 2 is supported.

- **Install a new OpenSSO 8.0 Update 2 deployment:** Install and configure the OpenSSO 8.0 Update 2 `opensso.war` file, as described in the *Sun OpenSSO Enterprise 8.0 Installation and Configuration Guide*.
- **Create a new specialized WAR file**: Use the `createwar` script to create one of the following new WAR files from the Update 2 `opensso.war` file:
  - OpenSSO Administration console only WAR
  - Distributed Authentication UI server WAR
  - OpenSSO server only WAR, without the Administration Console
  - IDP Discovery Service WAR

    For information, see Chapter 4, "Creating a Specialized OpenSSO Enterprise 8.0 Update 1 WAR File," in *Sun OpenSSO Enterprise 8.0 Update 1 Release Notes*.
- **Patch an existing specialized OpenSSO WAR file**: Use the `ssopatch` utility in Update 2 to patch an existing specialized OpenSSO 8.0 WAR file, as described inChapter 23, "Patching OpenSSO Enterprise 8.0," in *Sun OpenSSO Enterprise 8.0 Installation and Configuration Guide*

Note – If you are running Access Manager 7.1 or Access Manager 7 2005Q4 and you want to upgrade to Update 2, follow these steps:

1. Upgrade Access Manager 7.x to OpenSSO 8.0, as described in *Sun OpenSSO Enterprise 8.0 Upgrade Guide*.
2. Apply the Update 2 patch, as described in this chapter.

OpenSSO 8.0 Update 2 Patches

Sun periodically releases patches for OpenSSO 8.0 Update 2. For information about these patches, check back here periodically.

Planning Your Patch Operation

▼ To Plan Your Patch Operation for OpenSSO 8.0

1 **Read the "Overview of the `ssopatch` Utility" on page 41.**

2 **Install the patch utility for your platform, as described in "Installing the `ssopatch` Utility" on page 42.**

3. **Get information about your existing WAR file, to determine if your existing WAR file has been customized or modified, as described in "Comparing an OpenSSO WAR File to Its Internal Manifest" on page 44.**

4. **Compare your existing WAR file and the Update 2 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, as described in "Comparing Two OpenSSO WAR Files" on page 45.**

5. **Backup and archive your existing Opensso WAR file, as described in "Backing Up an OpenSSO WAR File" on page 43.**

6. **Patch your OpenSSO WAR File, as described in "Patching an OpenSSO WAR File" on page 45.**

7. **Run the `updateschema` script, as described in "Running the `updateschema` Script" on page 49.**

   Note - If you are patching a specialized WAR file that you generated from an `opensso.war`, such as an OpenSSO server only, administration console only, Distributed Authentication UI server, or IDP Discovery Service WAR, see "Patching a Specialized OpenSSO WAR" on page 48.

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 as `ssopatch.bat`.

Note - The syntax for `ssopatch` in OpenSSO 8.0 Update 2 has changed considerably since the OpenSSO 8.0 release. For the new syntax, see "Running the `updateschema` Script" on page 49.

The `ssopatch` patch utility performs these functions:

- Compares an OpenSSO WAR to its original manifest, to determine if the WAR file has been customized or modified
- Compare two OpenSSO 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 WAR file

After you download and unzip the OpenSSO 8.0 Update 2 ZIP file (`oracle_opensso_80U2.zip`), the patch utilities and related files are available in the `ssoPatchTools.zip` file, in the *zip-root*`/opensso/tools` directory, where *zip-root* is where you unzipped `oracle_opensso_80U2.zip`.

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

- A string that identifies the specific version of the OpenSSO WAR file

- All of the individual files in the OpenSSO 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 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.

Installing the `ssopatch` Utility

Before you install the `ssopatch` utility:

- Download and unzip the OpenSSO 8.0 Update 2 ZIP file (`oracle_opensso_80U2.zip`).
- Set your `JAVA_HOME` environment variable point to JDK 1.5 or later.

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 `oracle_opensso_80U2.zip`.
2. Create a new directory to unzip the `ssoPatchTools.zip` file. For example: `ssopatchtools`
3. Unzip the `ssoPatchTools.zip` file in the new directory.
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.

The following table describes the files in `ssoPatchTools.zip`.

File or Directory	Description
`README`	Readme file that describes `ssopatch`
`/lib`	Required `ssopatch` JAR files
`/patch`	`updateschema` and `updateschema.bat` scripts and related XML files
`/resources`	Required properties files
`ssopatch` and `ssopatch.bat`	Utilities for Solaris, Linux, and Windows systems

Backing Up an OpenSSO WAR File

Before you begin, backup your existing OpenSSO WAR file and configuration data:

- Copy your existing OpenSSO WAR file to a safe location. Then, if you need to back out Update 2 for some reason, you can re-deploy your backup copy of the WAR file.
- Backup your configuration data, as described in Chapter 15, "Backing Up and Restoring Configuration Data," in *Sun OpenSSO Enterprise 8.0 Administration Guide*.

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:

- `-war-file|-o` specifies a path to a WAR file (such as `opensso.war`) that has previously been deployed.
- `-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` species a path to a WAR file to compare against against the WAR file indicated by `-war-file|-o`.
- `-staging|-s` specifies a path to the staging area where the files from an OpenSSO 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.

Default is false (revision checking is performed).

- `-overwrite|-w` overwrites the files in the existing staging area. Default is false (files are not overwritten).

Comparing an OpenSSO WAR File to Its Internal Manifest

Use this procedure to determine if an OpenSSO WAR file has been customized or modified since it was downloaded.

The `ssopatch` utility generates a new internal manifest file and then compares this internal manifest against the manifest stored inside the original OpenSSO WAR file in the `META-INF` directory.

To Compare an OpenSSO WAR File to Its Internal Manifest

1. Run `ssopatch` to compare the OpenSSO WAR file to its internal manifest. For example:

```
./ssopatch -o /zip-root/opensso/deployable-war/opensso.war
Generating Manifest for: /zip-root/opensso/deployable-war/opensso.war
Comparing manifest of Internal (Enterprise 8.0 Build 6(200810311055))
against /zip-root/opensso/deployable-war/opensso.war (generated-200905050855)
File not in original war (images/login-origimage.jpg)
File updated in new war (images/login-backimage.jpg)
File updated in new war (WEB-INF/classes/amConfigurator.properties)
Differences: 3
```

This example shows these changes to the original WAR file:

- `images/login-origimage.jpg` is in `opensso.war` but was not found in the original manifest.
- `images/login-backimage.jpg` has been customized in `opensso.war` from the original manifest.
- `WEB-INF/classes/amConfigurator.properties` file has been customized in `opensso.war` from the original manifest.

Comparing Two OpenSSO WAR Files

Use this procedure to compare two WAR files, to show the files that have been:

- Customized in an original OpenSSO WAR
- Updated in a new OpenSSO WAR file
- Added or deleted between the two OpenSSO WAR versions

To Compare Two OpenSSO WAR Files

1. Run `ssopatch` to compare the two WAR files. In the example, the `-override` option is used to override the revision checking between the two WAR files:

```
./ssopatch -o /zip-root/opensso/deployable-war/opensso.war
-c /u1/opensso/deployable-war/opensso.war --override
Generating Manifest for: /zip-root/opensso/deployable-war/opensso.war
Original manifest: Enterprise 8.0 Build 6(200810311055)
New manifest: Enterprise 8.0 Update 2 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-200905050919) against
    /u1/opensso/deployable-war/opensso.war (generated-200905050920)
File updated in new war(WEB-INF/classes/amClientDetection_en.properties)
File updated in new war(WEB-INF/classes/fmSAMLConfiguration_fr.properties)
...
Differences: 1821
Customizations: 3
```

This example shows the files that have been updated and customized in the new WAR file.

Patching an OpenSSO WAR File

Use this procedure to create a new staging area, where an original WAR file is merged with a new WAR file.

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

- Files customized in the original WAR file
- Files updated in a new WAR file
- Files added or removed between the two WAR file versions

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

To Create a Staging Area to Patch an OpenSSO WAR File

1. Although the `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.
2. 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 2 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.

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

Use the following table to determine the action you might need to take for each file before you generate a new patched WAR file.

`ssopatch` Results	**Explanation and Action Required**
`File not in original war` *filename*	The indicated file does not exist in the original WAR file but is in the latest version of the WAR file. **Action**: None
`File updated in new war` *filename*	The indicated file exists in both the original and new WAR files and has been updated in the latest version of the WAR file. No customizations have been done in the original WAR file. **Action**: None
`File customized` *filename*	The indicated file exists in both WAR files, has been customized in the original version of the WAR file, but has not been updated in the latest version of the WAR file. **Action**: None

ssopatch Results	Explanation and Action Required
`May require manual customization` *filename*	The file exists in both WAR files, has been customized in the original version of the WAR file, and has been updated in the latest version of the WAR file. **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.

Next Steps

1. Create a new OpenSSO 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 new patched OpenSSO WAR file
2. Redeploy the `/patched/opensso.war` file to the web container using the original deploy URI. For example, `/opensso`

OpenSSO configuration changes. A new OpenSSO WAR file might have configuration changes that were not in your original WAR file. Any configuration changes, if any, will be documented separately for each patch. Check the patch documentation and the *Sun OpenSSO Enterprise 8.0 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.)

If you need to back out your patched version, undeploy the patched WAR file and then redeploy your original WAR file.

Creating an OpenSSO WAR Manifest File

An OpenSSO manifest file is a text file that identifies all of the individual files in a WAR file for a specific release, with checksum information for each file.

Use this procedure to create a manifest file that you can include in a specialized OpenSSO WAR, such as an OpenSSO server only, administration console only, Distributed Authentication UI server, or IDP Discovery Service WAR

To Create an OpenSSO WAR Manifest File

1. Run `ssopatch` to create the OpenSSO manifest file. For example:

   ```
   ./ssopatch -o zip-root/opensso/deployable-war/opensso.war --manifest /tmp/manifest
   ```

   where `opensso.war` is an existing OpenSSO WAR file.

   The `ssopatch` utility creates a new manifest file named `manifest` in the the `/tmp` directory.
2. To allow the WAR file to be patched, copy this new manifest file to the `META-INF` directory inside the `opensso.war` file. For example:

   ```
   mkdir META-INF
   cp /tmp/manifest META-INF
   jar uf opensso.war META-INF/manifest
   ```

Patching a Specialized OpenSSO WAR

If you have previously created a specialized OpenSSO WAR, such as an OpenSSO server only, administration console only, Distributed Authentication UI server, or IDP Discovery Service WAR, you can patch it by using the `ssopatch` utility.

▼ To Patch a Specialized OpenSSO WAR

Before You Begin The existing specialized WAR file and the OpenSSO 8.0 update 2 specialized WAR file should already be created.

Note – In the following example, the directory `zip-root` is the root directory for the unzipped contents of the currently deployed WAR file. The directory `/u2` is the root directory for the unzipped contents of the upgraded version that will be deployed.

---

1 Create a manifest file for the existing specialized OpenSSO WAR.

a. Run `ssopatch` to create the OpenSSO manifest file.

Example:

```
# cd /u2/opensso/tools/patch
# ./ssopatch -o zip-root/opensso/deployable-war/distauth.war --manifest
/tmp/OpenSSO.manifest
```

where `opensso.war` is an existing OpenSSO WAR file. The `ssopatch` utility creates a new manifest file named `manifest` in the `/tmp` directory.

b. To allow the WAR file to be patched, copy this new manifest file to the META-INF directory inside the `opensso.war` file.

Example:

```
# cd zip-root/opensso/deployable-war
# mkdir META-INF
# cp /tmp/OpenSSO.manifest META-INF
# jar uf distauth.war META-INF/OpenSSO.manifest
# rm -rf /tmp/OpenSSO.manifest
```

2 Generate a manifest file for the updated specialized WAR file.

Example:

```
# cd /u2/opensso/tools/patch

# ./ssopatch -o /u2/opensso/deployable-war/distauth.war
--manifest /tmp/OpenSSO.manifest
cd ../../deployable-war

# mkdir META-INF
# cp /tmp/OpenSSO.manifest META-INF
# jar uf distauth.war META-INF/OpenSSO.manifest
# rm -rf /tmp/OpenSSO.manifest
```

3 Use the `ssopatch` utility to compare your old and new WAR files.

Example:

```
# cd /u2/opensso/tools/patch
# ./ssopatch -o zip-root/opensso/deployable-war/distauth.war
-c /u2/opensso/deployable-war/distauth.war -override
```

4 Generate a staging area for the new specialized WAR file.

Example:

```
# cd /tmp/customized_staging
# jar cvf /patched/distauth.war *
```

5 Redeploy the /patched/distauth.war file to the web container using the original deploy URI.

Example, `/distauth`.

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 server version, adds new default server properties, adds new attribute schemas required for bug fixes and enhancements in Update 2. You must run `updateschema` in order to update the server version.

Before You Begin

- The `updateschema.sh` or `updateschema.bat` script requires the Update 2 version (or later) of the `ssoadm` command-line utility. Therefore, before you run this script, install the Update 2 admin tools, as described in Chapter 3, "Installing the OpenSSO Enterprise 8.0 Update 1 Admin Tools," in *Sun OpenSSO Enterprise 8.0 Update 1 Release Notes*.
- The `updateschema.bat` script executes several `ssoadm` commands. Therefore, before you run `updateschema.bat` on Windows systems, create a password file that contains the password user in clear text for the `amadmin` user. The `updateschema.bat` script prompts you for the path to the password file. Before the script terminates, it removes the password file.

To Run the `updateschema` Script

1. Change to the *patch-tools*`/patch` directory, where *patch-tools* is where you unzipped `ssoPatchTools.zip`.
2. Run `updateschema.sh` or `updateschema.bat`. For example, on Solaris systems:

   ```
   ./updateschema.sh
   ```

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.
4. Restart the OpenSSO 8.0 Update 2 web container.

Backing Out a Patch Installation

If you need to back out your patch installation, simply redeploy the original `opensso.war` file (or specialized WAR file).

◆ ◆ ◆ CHAPTER 4

Using the Security Token Service

As a trusted authority service, the OpenSSO Security Token Service issues and validates security tokens. As a web services security provider, the Security Token Service secures communication between the Web Service Client and the OpenSSO STS service itself. Many enhancements have been made to the Security Token Service since OpenSSO 8.0 Update 2.

This chapter contains the following topics:

Adding a WSSAuth Authentication Module

The Web Service Security authentication module enables OpenSSO to validate a UserName with a digest password received as an authentication token and contained in a service request from the web service client to a web service provider.

▼ To Add a New Web Service Security Authentication Module Instance

1. **In the OpenSSO console, go to the Access Control tab > *RealmName* > Authentication subtab.**
2. **In the Module Instances section, click New.**
3. **In the New Module Instance page, In the Name field, type a name for this WSSAuth authentication module instance.**
4. **For Type, choose WSSAuth.**

5 **Click OK.**

6 **Configure the WSSAuth authentication module instance.**

▼ To Configure a WSSAuth Authentication Module Instance

1 **In the OpenSSO console, go to the Access Control tab > *RealmName* > Authentication subtab.**

2 **In the Module Instances section, click name of the WSSAuth authentication module instance you want to configure.**

3 **Provide values for the WSSAuth Authentication Module Instance Realm attributes.**

The following table provides a listing and descriptions of the attributes you can configure.

User search attribute	Specify a user attribute that to be used to search for a user. Examples: `uid`, `cn`
User realm	Specify the realm the user belongs to. For OpenSSO STS it is always root realm, indicated by a forward slash / .
User password attribute	Specify a password attribute (password equivalent) for the user. The default could be `userpassword`, it could as well be `empoyeenumber` or `mail`.
Authentication Level	Specify a value that indicates how much to trust an authentication mechanism. The default value is `0`. The authentication level is set separately for each method of authentication. Once a user has authenticated, this value is stored in the `SSOToken` for the session. When the `SSOToken` is presented to an application the user wants to access, the application uses the stored value to determine whether the level is sufficient to grant the user access. If the authentication level stored in an `SSOToken` does not meet the minimum value required, the application can prompt the user to authenticate again through a service with a higher authentication level. `0` is a low value. For example, if the user accesses the URL *protocol*`://`*openssoServer:port*`/opensso/UI/Loin?authlevel=0`, a selection menu is displayed containing all authentication module instances with an authentication level of 0 or greater, or all authentication module instances. Similarly if the user accesses the

URL *protocol*`://`*openssoServer:port*`/opensso/UI/Loin?authlevel=50`, a selection menu is displayed containing authentication module instances with an authentication level of `50` or greater. Or if only one authentication module instance meets that constraint, a login screen for that authentication module instance is displayed.

If no authentication level is specified, the SSO token stores the value specified in the Core Authentication attribute Default Authentication Level.

Adding an OAMAuth Authentication Module

The Oracle authentication module enables OpenSSO to authenticate and single sign-on an administrator, who previously authenticated to Oracle Access Manager, to OpenSSO. The administrator does not have to provide credentials to OpenSSO.

▼ To Add a New Oracle Authentication Module Instance

1. **In the OpenSSO console, go to the Access Control tab > RealmName > Authentication subtab.**
2. **In the Module Instances section, click New.**
3. **In the Name field, type a name for this Oracle authentication module instance.**
4. **For Type, choose OAMAuth.**
5. **Click OK.**
6. **Configure the OAMAuth authentication module instance.**

▼ To Configure an Oracle Authentication Module Instance

1. **In the OpenSSO console, go to the Access Control tab > RealmName > Authentication subtab.**
2. **In the Module Instances section, click name of the OAMAuth authentication module instance you want to configure.**

3 Provide values for the Oracle Authentication Module Instance Realm attributes.

The following table provides a listing and descriptions of the attributes you can configure.

Remote User HeaderName	Specify the name of the `REMOTE USER HEADER` that is set by the Oracle Access Manager. Example: `OAM_REMOTE_USER`
Allowed user values	The Current Values list displays users who are allowed to access the OpenSSO STS administration console. ■ To add a user to the list, in the New Value field type a username, and then click Add. ■ To remove an entry from the Current Values list, select the entry and then click Remove.
Authentication level	Specify a value that indicates how much to trust an authentication mechanism. The default value is `0`. The authentication level is set separately for each method of authentication. Once a user has authenticated, this value is stored in the `SSOToken` for the session. When the `SSOToken` is presented to an application the user wants to access, the application uses the stored value to determine whether the level is sufficient to grant the user access. If the authentication level stored in an `SSOToken` does not meet the minimum value required, the application can prompt the user to authenticate again through a service with a higher authentication level. `0` is a low value. For example, if the user accesses the URL *protocol*`://`*openssoServer:port*`/opensso/UI/Loin?authlevel=0`, a selection menu is displayed containing all authentication module instances with an authentication level of 0 or greater, or all authentication module instances. Similarly if the user accesses the URL *protocol*`://`*openssoServer:port*`/opensso/UI/Loin?authlevel=50`, a selection menu is displayed containing authentication module instances with an authentication level of `50` or greater. Or if only one authentication module instance meets that constraint, a login screen for that authentication module instance is displayed. If no authentication level is specified, the SSO token stores the value specified in the Core Authentication attribute Default Authentication Level.

Generating Security Tokens

Oracle OpenSSO Security Token Service (OpenSSO STS) establishes a trust relationship between a web service client and a web service provider, and then brokers the trust between them. The web service can trust tokens issued by just one entity instead of having to communicate with several clients. In this way, OpenSSO STS significantly reduces trustpoint management overhead.

The following sections provide instructions for determining your security token needs, and for configuring the Security Token Service to generate and validate security tokens to meet those needs.

Registering a Web Service Provider to OpenSSO STS

When you add a new web service provider security agent profile, the web service provider is automatically registered to OpenSSO STS. See “To Create a New Agent Profile” in *Sun OpenSSO Enterprise 8.0 Administration Guide*.

Once you've registered a web service provider to OpenSSO STS, you can configure OpenSSO STS to generate web client security tokens acceptable by the web service provider.

Requesting a Web Service Client Security Token from OpenSSO STS

First determine what kind of security token the web service provider requires. OpenSSO STS supports Liberty Alliance Project Security Tokens and Web Services-Interoperability Basic Security Profile Security Tokens.

Using the Security Token Generation Matrix

Use the Security Token Generation Matrix to help you configure OpenSSO STS to generate a web service client security token required by the web service provider. First, in the last column titled OpenSSO STS Output Token, find a description that meets the web service provider token requirements. Then use the parameter values in the same row when you configure the Security Token Service. The "Token Generation Matrix Legend" provides information about the table headings and available options. See Section 5.2.3, "To Configure the Security Token Service" for detailed configuration instructions. For general information about Web Service Security and related terminology, see:

- http://www.oracle.com/technology/tech/standards/pdf/security.pdf

- http://download.oracle.com/docs/cd/E15523_01/web.1111/b32511/intro_security.htm#CDDHHGEE

The Security Token Generation Matrix summarizes frequently-used Security Token Service parameter settings and the types of security tokens OpenSSO STS generates based on these settings.

TABLE 4–1 Security Token Generation Matrix

Row	Message-Level Security Binding	Web Service Client Token	KeyType	OnBehalfOf Token	Use Key	OpenSSO STS Output Token
1	Asymmetric	X509	Bearer	Yes	No	SAML Bearer, no proof key
2	Asymmetric	Username	Bearer	Yes	No	SAML Bearer, no proof key
3	Asymmetric	X509	Bearer	No	No	SAML Bearer, no proof key
4	Asymmetric	Username	Bearer	No	No	SAML Bearer, no proof key
5	Asymmetric	X509	Symmetric	Yes	No	SAML Holder-of-Key, Symmetric proof key
6	Asymmetric	Username	Symmetric	Yes	No	SAML Holder-of-Key, Symmetric proof key
7	Asymmetric	X509	Symmetric	No	No	SAML Holder-of-Key, Symme
8	Asymmetric	Username	Symmetric	No	No	SAML Holder-of-Key, Symmetric proof key
9	Asymmetric	X509	Asymmetric	No	Web Service Client public key	SAML Holder-of-Key, Asymmetric proof key

TABLE 4–1 Security Token Generation Matrix *(Continued)*

10	Asymmetric	X509	Oracle-proprietary for SAML sender-vouches	Yes	No	SAML sender-vouches, no proof key
11	Asymmetric	Username	Oracle-proprietary for SAML sender-vouches	Yes	No	SAML sender-vouches, no proof key
12	Transport	Username	Bearer	Yes	No	SAML Bearer, no proof key
13	Transport	Username	Bearer	No	No	SAML Bearer, no proof key
14	Transport	Username	Symmetric	Yes	No	SAML Holder-of-Key, Symmetric
15	Transport	Username	Symmetric	No	No	SAML Holder-of-Key, Symmetric proof key
16	Transport	Username	Oracle-proprietary for SAML sender-vouches	Yes	No	SAML sender-vouches, no proof key
17	Asymmetric	X509	Asymmetric	No	No	SAML Holder-of-Key, Asymmetric proof key
18	Asymmetric	X509	No	No	No	SAML Holder-of-Key, Asymmetric proof key
19	Asymmetric	Username	No	No	No	SAML Holder-of-Key, Symmetric proof key
20	Transport	Username	No	No	No	SAML Holder-of-Key, Symmetric proof key

◆ ◆ ◆ CHAPTER 5

Using the Oracle OpenSSO Fedlet

This section provides the following information about the Oracle OpenSSO Fedlet:

About the Oracle OpenSSO Fedlet

The Oracle OpenSSO Fedlet is a lightweight service provider (SP) implementation that can be deployed with a Java or .NET service provider application, enabling the application to communicate with an identity provider (IDP) such as Oracle OpenSSO 8.0 Update 2 using the SAMLv2 protocol. The Fedlet has two versions, depending on your platform:

- The Java Fedlet was first released in OpenSSO 8.0. For information, see Chapter 5, "Using the OpenSSO Enterprise Fedlet to Enable Identity Federation," in *Sun OpenSSO Enterprise 8.0 Deployment Planning Guide*.
- The .NET Fedlet was released in OpenSSO 8.0 Update 1. For information, see Chapter 10, "Using the ASP.NET Fedlet with OpenSSO Enterprise 8.0 Update 1," in *Sun OpenSSO Enterprise 8.0 Update 1 Release Notes*.

In Oracle OpenSSO 8.0 Update 2, the Fedlet is available as follows:

- After you unzip the OpenSSO 8.0 Update 2 ZIP file, both the Java Fedlet and .NET Fedlet are available in the following file:

  zip-root`/opensso/fedlet/fedlet-unconfigured.zip`, where *zip-root* is where you unzipped the Oracle OpenSSO 8.0 Update 2 ZIP file.
- After you install Oracle OpenSSO 8.0 Update 2, you can create the Java Fedlet in the OpenSSO 8.0 Administration Console using the Create Fedlet work flow under Common Tasks.

Requirements for the Oracle OpenSSO Fedlet

The Fedlet has the following requirements:

- Oracle OpenSSO 8.0 Update 2 supported web container, if you plan to deploy the `fedlet.war`, or a Java service provider application that is integrated with the Fedlet. See the "Hardware and Software Requirements For OpenSSO 8.0 Update 2" on page 14.
- Microsoft Internet Information Server (IIS) 7.0 and later, if you plan to deploy the .NET Fedlet
- JDK 1.6.x and later

Oracle OpenSSO Fedlet Configuration

This section describes how to initially configure the Fedlet with a service provider application:

- "To Configure the Java Fedlet" on page 60
- "To Configure the .NET Fedlet" on page 62

After you finish the initial configuration for the Fedlet, continue with any additional configuration you want to perform. Several considerations are:

- If you modify the Fedlet `sp.xml` file, you must re-import this file into your identity provider.
- If you make other Fedlet configuration changes on the service provider side, convey this information to the identity provider administrator, so that the required configuration changes can be made on the identity provider side.

▼ To Configure the Java Fedlet

1. **On the identity provider side, generate the XML metadata for the identity provider and save the metadata in a file named `idp.xml`.**

   For Oracle OpenSSO 8.0 Update 2, use `exportmetadata.jsp`. For example:

   ```
   http://opensso-idp.example.com:8080/opensso/saml2/jsp/exportmetadata.jsp
   ```

2. **On the service provider side, unzip the Fedlet ZIP file (if necessary).**

3. **Create the Fedlet home directory, which is the directory where the Fedlet reads its metadata, circle of trust, and configuration properties files.**

   The default location is the fedlet subdirectory under the home directory of the user running the Fedlet web container (indicated by the `user.home` JVM property). For example, if this home directory is `/home/webservd`, the Fedlet home directory is:

   ```
   /home/webservd/fedlet
   ```

To change the Fedlet default home directory, set the value of the JVM run-time `com.sun.identity.fedlet.home` property to the desired location. For example:

```
-Dcom.sun.identity.fedlet.home=/export/fedlet/conf
```

The Fedlet then reads its metadata, circle of trust, and configuration files from the `/export/fedlet/conf` directory.

4 Copy the following files from the Java Fedlet `java/conf` directory to the Fedlet home directory:

- `sp.xml-template`
- `sp-extended.xml-template`
- `idp-extended.xml-template`
- `fedlet.cot-template`

5 In the Fedlet home directory, rename the files you copied and drop `-template` from each name.

6 In the files you copied and renamed in the Fedlet home directory, replace the tags as shown in the next table:

Tag	Replace With
`FEDLET_COT`	Name of the circle of trust (COT) of which the remote identity provider and the Java Fedlet service provider application are members.
`FEDLET_ENTITY_ID`	ID (name) of the Java Fedlet service provider application. For example: `fedletsp`
`FEDLET_PROTOCOL`	Protocol of the web container for the Java Fedlet service provider application (such as `fedlet.war`). For example: `https`
`FEDLET_HOST`	Host name of the web container for the Java Fedlet service provider application (such as `fedlet.war`). For example: `fedlet-host.example.com`
`FEDLET_PORT`	Port number of the web container for the Java Fedlet service provider application (such as `fedlet.war`). For example: `80`
`FEDLET_DEPLOY_URI`	URL of the Java Fedlet service provider application. For example: `http://fedletsp.example.com/myFedletApp`
`IDP_ENTITY_ID`	ID (name) of the remote identity provider. For example: `openssoidp`

Note: If the Fedlet service provider or identity provider entity ID contains a percent sign (%) or comma (,), you must escape the character before replacing it in the `fedlet.cot` file. For example, change "%" to "%25" and "," to "%2C".

7 Copy the `FedletConfiguration.properties` file from the Java Fedlet `java/conf` directory to the Fedlet home directory.

8. **Copy the identity provider standard metadata XML file (from Step 1) to the Fedlet home directory. This file must be named `idp.xml`.**

9. **Import the Java Fedlet XML metadata file (`sp.xml`) into the identity provider.**

   For Oracle OpenSSO 8.0 Update 2, use the Register Remote Service Provider work flow under Common Tasks in the OpenSSO 8.0 Administration Console to import the Java Fedlet service provider metadata and to add the Java Fedlet service provider to a circle of trust.

Next Steps Depending on your requirements, continue with any additional configuration for the Java Fedlet.

▼ To Configure the .NET Fedlet

1. **On the identity provider side, generate the XML metadata for the identity provider and save the metadata in a file named `idp.xml`.**

   For Oracle OpenSSO 8.0 Update 2, use `exportmetadata.jsp`. For example:

   ```
   http://opensso-idp.example.com:8080/opensso/saml2/jsp/exportmetadata.jsp
   ```

2. **On the service provider side, unzip the Fedlet ZIP file (if necessary).**

3. **Copy the following files from the .NET Fedlet `asp.net/conf` folder to your application's `App_Data` folder:**

   - `sp.xml-template`
   - `sp-extended.xml-template`
   - `idp-extended.xml-template`
   - `fedlet.cot-template`

4. **In the `App_Data` folder, rename the files you copied and drop `-template` from each name.**

5. **In the files you copied and renamed in the `App_Data` folder, replace the tags as shown in the next table:**

Tag	Replace With
`FEDLET_COT`	Name of the circle of trust (COT) of which the remote identity provider and the .NET Fedlet service provider application are members.
`FEDLET_ENTITY_ID`	ID (name) of the .NET Fedlet service provider application. For example: `fedletsp`
`FEDLET_DEPLOY_URI`	URL of the .NET Fedlet service provider application. For example: `http://fedletsp.example.com/myFedletApp`
`IDP_ENTITY_ID`	ID (name) of the remote identity provider. For example: `openssoidp`

6. **Copy the identity provider standard metadata XML file (from Step 1) to your application's `App_Data` folder. This file must be named `idp.xml`.**

7. **Copy the `Fedlet.dll` and the `Fedlet.dll.config` files from the .NET Fedlet `asp.net/bin` folder to the application's `bin` folder.**

8. **Import the .NET Fedlet XML metadata file (`sp.xml`) into the identity provider.**

   For Oracle OpenSSO 8.0 Update 2, use the Register Remote Service Provider work flow under Common Tasks in the OpenSSO 8.0 Administration Console to import the .NET Fedlet service provider metadata and to add the .NET Fedlet service provider to a circle of trust.

Next Steps Depending on your requirements, continue with any additional configuration for the .NET Fedlet.

New Features for the Fedlet in OpenSSO 8.0 Update 2

Oracle OpenSSO 8.0 Update 2 includes the following new features for the Fedlet:

Fedlet Version Information (CR 6941387)

The Oracle OpenSSO Fedlet includes version information. After you extract the files in the Fedlet package (ZIP file), determine the Fedlet version by viewing one of the following files:

- Java Fedlet: `java/conf/FederationConfig.properties`
- .NET Fedlet: `asp.net/bin/Fedlet.dll.config`

Java Fedlet Password Encryption and Decryption (CR 6930477)

The Java Fedlet provides the `fedletEncode.jsp` in the `fedlet.war` file to encrypt the `storepass` and `keypass` passwords. By default, a different encryption key is generated for each Fedlet. To change this encryption key, set the `am.encryption.pwd` property in the Fedlet `FederationConfig.properties` file.

Java Fedlet Support for Signing and Encryption

The Java Fedlet supports XML signature verification and decryption of encrypted `assertion` and `NameID` elements and their corresponding attributes.

▼ To Configure the Java Fedlet for Signing and Encryption

1. **Create a keystore file named `keystore.jks` using the `keytool` utility.**
2. **Add the private key (and public certificate if applicable) used for signing and the private key (and public certificate if applicable) used for encryption to the `keystore.jks` file.**
3. **Create a `.storepass` file.**
4. **Add the password to the `.storepass` file. To encrypt the password, use `fedletEncode.jsp`.**
5. **Create a `.keypass` file.**
6. **Add the password to the `.keypass` file. To encrypt the password, use `fedletEncode.jsp`.**
7. **If you are using clear text passwords, comment out the following line in the `FederationConfig.properties` file:**

   ```
   com.sun.identity.saml.xmlsig.passwordDecoder=
       com.sun.identity.fedlet.FedletEncodeDecode
   ```

8. **Set the complete path for the following attributes in the `FederationConfig.properties` file, where *path* is the complete path to the respective file:**

   ```
   com.sun.identity.saml.xmlsig.keystore=path/keystore.jks
   com.sun.identity.saml.xmlsig.storepass=path/.storepass
   com.sun.identity.saml.xmlsig.keypass=path/.keypass
   ```

9. **Use `keytool` to export the signing certificate. For example:**

   ```
   keytool -export -keystore keystore.jks -rfc -alias test
   ```

   The tool prompts you to enter the password used to access `keystore.jks` and then generates the certificate.

10 **If you need an encryption certificate, use `keytool` to export it, as shown in the previous step. (Or use the same certificate for both signing and encryption.)**

11 **Create a `KeyDescriptor` XML block and add the encryption certificate to it. For example, note the `use="signing"` tag of the `KeyDescriptor` element:**

```
<KeyDescriptor use="signing">
      <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:X509Data>
          <ds:X509Certificate>
MIICQDCCAakCBEeNB0swDQYJKoZIhvcNAQEEBQAwZzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNh
bGlmb3JuaWExFDASBgNVBAcTC1NhbnRhIENsYXJhMQwwCgYDVQQKEwNTdW4xEDAOBgNVBAsTB09w
ZW5TU08xDTALBgNVBAMTBHRlc3QwHhcNMDgwMTE1MTkxOTM5WhcNMTgwMTEyMTkxOTM5WjBnMQsw
CQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEUMBIGA1UEBxMLU2FudGEgQ2xhcmExDDAK
BgNVBAoTA1N1bjEQMA4GA1UECxMHT3BlblNTTzENMAsGA1UEAxMEdGVzdDCBnzANBgkqhkiG9w0B
AQEFAAOBjQAwgYkCgYEArSQc/U75GB2AtKhbGS5piiLkmJzqEsp64rDxbMJ+xDrye0EN/q1U5Of\+
RkDsaN/igkAvV1cuXEgTL6RlafFPcUX7QxDhZBhsYF9pbwtMzi4A4su9hnxIhURebGEmxKW9qJNY
Js0Vo5+IgjxuEWnjnnVgHTs1+mq5QYTA7E6ZyL8CAwEAATANBgkqhkiG9w0BAQQFAAOBgQB3Pw/U
QzPKTPTYi9upbFXlrAKMwtFf2OW4yvGWWvlcwcNSZJmTJ8ARvVYOMEVNbsT4OFcfu2/PeYoAdiDA
cGy/F2Zuj8XJJpuQRSE6PtQqBuDEHjjmOQJ0rV/r8mO1ZCtHRhpZ5zYRjhRC9eCbjx9VrFax0JDC
/FfwWigmrW0Y0Q==
          </ds:X509Certificate>
        </ds:X509Data>
      </ds:KeyInfo>
</KeyDescriptor>
```

12 **Create another `KeyDescriptor` XML block and add the encryption certificate to it. For example, note the `use="encryption"` tag of the `KeyDescriptor` element:**

```
<KeyDescriptor use="encryption">
      <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
        <X509Data>
          <X509Certificate>
MIICQDCCAakCBEeNB0swDQYJKoZIhvcNAQEEBQAwZzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNh
bGlmb3JuaWExFDASBgNVBAcTC1NhbnRhIENsYXJhMQwwCgYDVQQKEwNTdW4xEDAOBgNVBAsTB09w
ZW5TU08xDTALBgNVBAMTBHRlc3QwHhcNMDgwMTE1MTkxOTM5WhcNMTgwMTEyMTkxOTM5WjBnMQsw
CQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEUMBIGA1UEBxMLU2FudGEgQ2xhcmExDDAK
BgNVBAoTA1N1bjEQMA4GA1UECxMHT3BlblNTTzENMAsGA1UEAxMEdGVzdDCBnzANBgkqhkiG9w0B
AQEFAAOBjQAwgYkCgYEArSQc/U75GB2AtKhbGS5piiLkmJzqEsp64rDxbMJ+xDrye0EN/q1U5Of\+
RkDsaN/igkAvV1cuXEgTL6RlafFPcUX7QxDhZBhsYF9pbwtMzi4A4su9hnxIhURebGEmxKW9qJNY
Js0Vo5+IgjxuEWnjnnVgHTs1+mq5QYTA7E6ZyL8CAwEAATANBgkqhkiG9w0BAQQFAAOBgQB3Pw/U
QzPKTPTYi9upbFXlrAKMwtFf2OW4yvGWWvlcwcNSZJmTJ8ARvVYOMEVNbsT4OFcfu2/PeYoAdiDA
cGy/F2Zuj8XJJpuQRSE6PtQqBuDEHjjmOQJ0rV/r8mO1ZCtHRhpZ5zYRjhRC9eCbjx9VrFax0JDC
/FfwWigmrW0Y0Q==
          </X509Certificate>
        </X509Data>
      </KeyInfo>
<EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc">
  <KeySize xmlns="http://www.w3.org/2001/04/xmlenc#">128</KeySize>
</EncryptionMethod>
</KeyDescriptor>
```

13 **In the Java Fedlet `sp.xml` file, add the XML blocks with the signing and encryption certificates under the `SPSSODescriptor` element. For a sample `SPSSODescriptor` element, see Example 5–1.**

The `AuthnRequestsSigned` attribute is set to `true`, configuring the Java Fedlet to sign all authentication requests.

14 **In the Java Fedlet `sp-extended.xml` file, set values for the following elements:**

- `signingCertAlias` contains the alias of the XML signing certificate in the keystore.
- `encryptionCertAlias` contains the alias of the XML encryption certificate in the keystore.

15 **To enforce what the Java Fedlet service provider encrypts, set the following attributes in the `sp-extended.xml` file to `true`:**

- `wantAssertionEncrypted`
- `wantNameIDEncrypted`
- `wantAttributeEncrypted`

16 **To enforce what the Java Fedlet service provider signs and wants signed, set the following attributes to `true`:**

- `wantAuthnRequestsSigned` in the `idp.xml` file tells the Fedlet what to sign.
- `AuthnRequestsSigned` and `WantAssertionsSigned` in the `sp.xml` file tells the identity provider what the Fedlet plans to sign.
- `wantArtifactResponseSigned` in the `sp-extended.xml` file tells the Fedlet what to sign.
- `wantPOSTResponseSigned` in the `sp-extended.xml` file
- `wantLogoutRequestSigned` in the `sp-extended.xml` file
- `wantLogoutResponseSigned` in the `sp-extended.xml` file

If the identity provider requires signing for specific messages, set the respective attributes to `true` in the `idp-extended.xml` file. For example, `wantLogoutRequestSigned` and `wantLogoutResponseSigned`.

Note – If you set attributes in the `sp-extended.xml` file, convey this information to the identity provider administrator, so that the necessary configuration changes can be made in the identity provider.

---

17 **Restart the Java Fedlet web container.**

18 **Import the Java Fedlet `sp.xml` file into the identity provider.**

Example 5–1 Java Fedlet Sample `SPSSODescriptor` Element

```
<EntityDescriptor entityID="fedlet"
xmlns="urn:oasis:names:tc:SAML:2.0:metadata">

<SPSSODescriptor AuthnRequestsSigned="true" WantAssertionsSigned="false"
protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
<b><KeyDescriptor use="signing">
      <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:X509Data>
          <ds:X509Certificate>
MIICQDCCAakCBEeNB0swDQYJKoZIhvcNAQEEBQAwZzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNh
bGlmb3JuaWExFDASBgNVBAcTC1NhbnRhIENsYXJhMQwwCgYDVQQKEwNTdW4xEDAOBgNVBAsTB09w
ZW5TU08xDTALBgNVBAMTBHRlc3QwHhcNMDgwMTE1MTkxOTM5WhcNMTgwMTEyMTkxOTM5WjBnMQsw
CQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEUMBIGA1UEBxMLU2FudGEgQ2xhcmExDDAK
BgNVBAoTA1N1bjEQMA4GA1UECxMHT3BlblNTTzENMAsGA1UEAxMEdGVzdDBnzANBgkqhkiG9w0B
AQEFAAOBjQAwgYkCgYEArSQc/U75GB2AtKhbGS5piiLkmJzqEsp64rDxbMJ+xDrye0EN/q1U5Of\+
RkDsaN/igkAvV1cuXEgTL6RlafFPcUX7QxDhZBhsYF9pbwtMzi4A4su9hnxIhURebGEmxKW9qJNY
Js0Vo5+IgjxuEWnjnnVgHTs1+mq5QYTA7E6ZyL8CAwEAATANBgkqhkiG9w0BAQQFAAOBgQB3Pw/U
QzPKTPTYi9upbFXlrAKMwtFf2OW4yvGWWvlcwcNSZJmTJ8ARvVYOMEVNbsT4OFcfu2/PeYoAdiDA
cGy/F2Zuj8XJJpuQRSE6PtQqBuDEHjjmOQJ0rV/r8mO1ZCtHRhpZ5zYRjhRC9eCbjx9VrFax0JDC
/FfwWigmrW0Y0Q==
          </ds:X509Certificate>
        </ds:X509Data>
      </ds:KeyInfo>

</KeyDescriptor></b>
<b><KeyDescriptor use="encryption">
      <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
        <X509Data>
          <X509Certificate>
MIICQDCCAakCBEeNB0swDQYJKoZIhvcNAQEEBQAwZzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNh
bGlmb3JuaWExFDASBgNVBAcTC1NhbnRhIENsYXJhMQwwCgYDVQQKEwNTdW4xEDAOBgNVBAsTB09w
ZW5TU08xDTALBgNVBAMTBHRlc3QwHhcNMDgwMTE1MTkxOTM5WhcNMTgwMTEyMTkxOTM5WjBnMQsw
CQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEUMBIGA1UEBxMLU2FudGEgQ2xhcmExDDAK
BgNVBAoTA1N1bjEQMA4GA1UECxMHT3BlblNTTzENMAsGA1UEAxMEdGVzdDBnzANBgkqhkiG9w0B
AQEFAAOBjQAwgYkCgYEArSQc/U75GB2AtKhbGS5piiLkmJzqEsp64rDxbMJ+xDrye0EN/q1U5Of\+
RkDsaN/igkAvV1cuXEgTL6RlafFPcUX7QxDhZBhsYF9pbwtMzi4A4su9hnxIhURebGEmxKW9qJNY
Js0Vo5+IgjxuEWnjnnVgHTs1+mq5QYTA7E6ZyL8CAwEAATANBgkqhkiG9w0BAQQFAAOBgQB3Pw/U
QzPKTPTYi9upbFXlrAKMwtFf2OW4yvGWWvlcwcNSZJmTJ8ARvVYOMEVNbsT4OFcfu2/PeYoAdiDA
cGy/F2Zuj8XJJpuQRSE6PtQqBuDEHjjmOQJ0rV/r8mO1ZCtHRhpZ5zYRjhRC9eCbjx9VrFax0JDC
/FfwWigmrW0Y0Q==
          </X509Certificate>
        </X509Data>
      </KeyInfo>

<EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc">
<KeySize xmlns="http://www.w3.org/2001/04/xmlenc#">128</KeySize>
</EncryptionMethod>
</KeyDescriptor></b>
<NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat
><AssertionConsumerService index="1"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Location="http://server.sun.com:7070/fedlet/fedletapplication"/>
</SPSSODescriptor>
</EntityDescriptor>
```

Java Fedlet Support for Attribute Query (CR 6930476)

The Java Fedlet supports the SAMLv2 Attribute Query to query an identity provider such as Oracle OpenSSO 8.0 Update 2 for specific identity attribute values. You can configure the Fedlet to sign the query and encrypt the query. Signing is required for issuing a Fedlet query, but encryption is optional.

▼ To Configure the Java Fedlet for Attribute Query

1 **Enable XML signing to sign the Attribute Query, as described in "Java Fedlet Support for Signing and Encryption" on page 64.**

2 **Add the certificate generated in the previous step to the `RoleDescriptor` element in the Fedlet `sp.xml` file. In the following example, there are two `KeyDescriptor` tags in which you paste the certificate. One is for signing and another is for encryption. If you are not enabling encryption, the `KeyDescriptor use="encryption" tag` is not required.**

```
<RoleDescriptor xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:query="urn:oasis:names:tc:SAML:metadata:ext:query"
      xsi:type="query:AttributeQueryDescriptorType"
      protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
  <KeyDescriptor use="signing">
      <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:X509Data>
          <ds:X509Certificate>
            --certificate--
          </ds:X509Certificate>
        </ds:X509Data>
      </ds:KeyInfo>
  </KeyDescriptor>
  <KeyDescriptor use="encryption">
      <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:X509Data>
          <ds:X509Certificate>
            --certificate--
          </ds:X509Certificate>
        </ds:X509Data>
      </ds:KeyInfo>
    <EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc">
<xenc:KeySize
xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">128</xenc:KeySize>
    </EncryptionMethod>
  </KeyDescriptor>
</RoleDescriptor>
```

3. **In the Java Fedlet `sp-extended.xml` file, specify the value for the `signingCertAlias` attribute and if configured, for the `encryptionCertAlias` attribute.**

   If you plan to configure the identity provider to encrypt the assertion, also encrypt the `NameID` element. Thus, the value of the `wantNameIDEncrypted` attribute must be set to `true`. Add the XML code to the `AttributeQueryConfig` element. For example:

   ```
   <Attribute name="signingCertAlias">
         <Value>test</Value>
   </Attribute>
   <Attribute name="encryptionCertAlias">
         <Value>test</Value>
   </Attribute>
   <Attribute name="wantNameIDEncrypted">
         <Value>true</Value>
   </Attribute>
   ```

   In this example, `test` is the alias for the sample key.

4. **Import the Java Fedlet metadata file (`sp.xml`) into the identity provider.**

   Also, perform the additional configuration steps in the identity provider to support the Attribute Query for the Fedlet.

.NET Fedlet Encryption and Decryption of Requests and Responses (CR 6939005)

The .NET Fedlet can encrypt outgoing XML requests and decrypt incoming responses for the NameID, Attribute, and Assertion elements.

▼ To Configure the .NET Fedlet for Encryption and Decryption of Requests and Responses

1. **Import your X.509 certificate to the Personal folder within the Local Computer account using the Certificates Snap-in for the Microsoft Management Console. To use this snap-in, see the following Microsoft article:**

   `http://msdn.microsoft.com/en-us/library/ms788967.aspx`

2. **Specify a friendly name for this certificate by viewing the Properties dialog and entering a value. (Save this value for Step 4.)**

3. **Set the appropriate permissions to allow read access to the certificate for the user account used by Internet Information Server (IIS) as described at the Microsoft article. For example:**

   a. **In the Certificates Snap-in, navigate to Action, All Tasks, and then Manage Private Keys.**

b. Specify Allow Read permissions for the user account running IIS (usually NETWORK SERVICE).

4 In the .NET Fedlet's extended metadata file (`sp-extended.xml`), specify the friendly name specified in Step 2 as the value for the `encryptionCertAlias` attribute. For example:

```
<Attribute name="encryptionCertAlias">
<Value>MyFedlet</Value>
```

5 In the .NET Fedlet's service provider metadata file (`sp.xml`), add the `KeyDescriptor` for the encryption key.

Use the Certificates Snap-in for the Microsoft Management Console used earlier to export the public key of your certificate in Base64 encoding to be included in the `KeyDescriptor` XML block. This `KeyDescriptor` must be the first child element within the `SPSSODescriptor`. For example:

```
<KeyDescriptor use="encryption">
           <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
             <ds:X509Data>
               <ds:X509Certificate>
MIICQDCCAakCBEeNB0swDQYJKoZIhvcNAQEEBQAwZzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNh
bGlmb3JuaWExFDASBgNVBAcTC1NhbnRhIENsYXJhMQwwCgYDVQQKEwNTdW4xEDAOBgNVBAsTB09w
ZW5TU08xDTALBgNVBAMTBHRlc3QwHhcNMDgwMTE1MTkxOTM5WhcNMTgwMTEyMTkxOTM5WjBnMQsw
CQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEUMBIGA1UEBxMLU2FudGEgQ2xhcmExDDAK
BgNVBAoTA1N1bjEQMA4GA1UECxMHT3BlblNTTzENMAsGA1UEAxMEdGVzdDCBnzANBgkqhkiG9w0B
AQEFAAOBjQAwgYkCgYEArSQc/U75GB2AtKhbGS5piiLkmJzqEsp64rDxbMJ+xDrye0EN/q1U5Of\+
RkDsaN/igkAvV1cuXEgTL6RlafFPcUX7QxDhZBhsYF9pbwtMzi4A4su9hnxIhURebGEmxKW9qJNY
Js0Vo5+IgjxuEWnjnnVgHTs1+mq5QYTA7E6ZyL8CAwEAATANBgkqhkiG9w0BAQQFAAOBgQB3Pw/U
QzPKTPTYi9upbFXlrAKMwtFf2OW4yvGWWvlcwcNSZJmTJ8ARvVYOMEVNbsT4OFcfu2/PeYoAdiDA
cGy/F2Zuj8XJJpuQRSE6PtQqBuDEHjjmOQJ0rV/r8mO1ZCtHRhpZ5zYRjhRC9eCbjx9VrFax0JDC
/FfwWigmrW0Y0Q==
               </ds:X509Certificate>
             </ds:X509Data>
           </ds:KeyInfo>
           <EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc">
           <KeySize
xmlns="http://www.w3.org/2001/04/xmlenc#">128</KeySize>
           </EncryptionMethod>
</KeyDescriptor>
```

6 Restart the Application Pool associated with your .NET application.

Next Steps To test this configuration, use the sample application. Also, set the following attributes to encrypt requests and decrypt responses with the identity provider with the appropriate changes to the configured metadata:

- Assertion: Set the `wantAssertionEncrypted` attribute in the `sp-extended.xml` metadata file to `true` to have the .NET Fedlet decrypt the `EncryptedAssertion` element in incoming responses from the identity provider.

- Attribute: Set the `wantAttributeEncrypted` attribute in the `sp-extended.xml` metadata `file` to true to have the .NET Fedlet decrypt the `EncryptedAttribute` element in incoming responses from the identity provider.
- NameID: Set the `wantNameIDEncrypted` attribute in the `idp-extended.xml` metadata file to `true` to have the .NET Fedlet encrypt the `NameID` element in outgoing requests. Set this same attribute in `sp-extended.xml` to have the .NET Fedlet decrypt the `EncryptedID` element in incoming responses from the identity provider.

.NET Fedlet Signing of Requests and Responses (CR 6928530)

The .NET Fedlet supports the signing of outgoing XML requests such as `Authn` requests and logout requests.

▼ To Configure the .NET Fedlet for Signing of Requests and Responses:

1. **Import your X.509 certificate to the Personal folder within the Local Computer account using the Certificates Snap-in for the Microsoft Management Console. To use this snap-in, see the following Microsoft article:**

   `http://msdn.microsoft.com/en-us/library/ms788967.aspx`

2. **Specify a friendly name for this certificate by viewing the Properties dialog and entering a value. (Save this value for Step 4.)**

3. **Set the appropriate permissions to allow read access to the certificate for the user account used by Internet Information Server (IIS) as described at the Microsoft article. For example:**

   a. **In the Certificates Snap-in, navigate to Action, All Tasks, and then Manage Private Keys.**

   b. **Specify Allow Read permissions for the user account running IIS (usually NETWORK SERVICE).**

4. **In the .NET Fedlet's extended metadata file (`sp-extended.xml`), specify the friendly name specified in Step 2 as the value for the `signingCertAlias` attribute. For example:**

```
<Attribute name="signingCertAlias">
<Value>MyFedlet</Value>
```

5 **In the .NET Fedlet's service provider metadata file (`sp.xml`), add the `KeyDescriptor` for the signing key.**

Use the Certificates Snap-in for the Microsoft Management Console used earlier to export the public key of your certificate in Base64 encoding to be included in the `KeyDescriptor` XML block. This `KeyDescriptor` must be the first child element within the `SPSSODescriptor`. For example:

```
<KeyDescriptor use="signing">
             <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
                <ds:X509Data>
                   <ds:X509Certificate>
MIICQDCCAakCBEeNB0swDQYJKoZIhvcNAQEEBQAwZzELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNh
bGlmb3JuaWExFDASBgNVBAcTC1NhbnRhIENsYXJhMQwwCgYDVQQKEwNTdW4xEDAOBgNVBAsTB09w
ZW5TU08xDTALBgNVBAMTBHRlc3QwHhcNMDgwMTE1MTkxOTM5WhcNMTgwMTEyMTkxOTM5WjBnMQsw
CQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEUMBIGA1UEBxMLU2FudGEgQ2xhcmExDDAK
BgNVBAoTA1N1bjEQMA4GA1UECxMHT3BlblNTTzENMAsGA1UEAxMEdGVzdDCBnzANBgkqhkiG9w0B
AQEFAAOBjQAwgYkCgYEArSQc/U75GB2AtKhbGS5piiLkmJzqEsp64rDxbMJ+xDrye0EN/q1U5Of\+
RkDsaN/igkAvV1cuXEgTL6RlafFPcUX7QxDhZBhsYF9pbwtMzi4A4su9hnxIhURebGEmxKW9qJNY
Js0Vo5+IgjxuEWnjnnVgHTs1+mq5QYTA7E6ZyL8CAwEAATANBgkqhkiG9w0BAQQFAAOBgQB3Pw/U
QzPKTPTYi9upbFXlrAKMwtFf2OW4yvGWWvlcwcNSZJmTJ8ARvVYOMEVNbsT4OFcfu2/PeYoAdiDA
cGy/F2Zuj8XJJpuQRSE6PtQqBuDEHjjmOQJ0rV/r8mO1ZCtHRhpZ5zYRjhRC9eCbjx9VrFax0JDC
/FfwWigmrW0Y0Q==
                   </ds:X509Certificate>
                </ds:X509Data>
             </ds:KeyInfo>
</KeyDescriptor>
```

6 **Restart the Application Pool associated with your .NET application.**

.NET Fedlet Single Logout (CR 6928528 and CR 6930472)

The .NET Fedlet supports both identity provider initiated and service provider initiated single logout. To implement single logout, the .NET Fedlet sample application includes the `logout.aspx` and `spinitiatedslo.aspx` files in the `asp.net/SampleApp` folder. To see how the Fedlet single logout feature works, deploy the .NET Fedlet sample application.

▼ To Configure a .NET Fedlet Service Provider Application for Single Logout:

1 **If you have not configured the .NET Fedlet, follow the steps in the `Readme` file.**

2 **Copy the `logout.aspx` and `spinitiatedslo.aspx` files within your .NET application's public content.**

3 **Make these changes to the configuration files for your application:**

- In the `sp.xml` file, make sure the path to the `logout.aspx` file points to the correct location of the file for your application.
- In the `idp.xml` file (or during the identity provider configuration) make sure the path to the `spinitiatedslo.aspx` file points to the correct location of the file for your application.

4 **If you want the logout request and logout response signed, set the following attributes to `true` in the `sp-extended.xml` and `idp-extended.xml` files:**

- `wantLogoutRequestSigned`
- `wantLogoutResponseSigned`

5 **Import the Fedlet service provider metadata file (`sp.xml`) into the identity provider.**

Also, inform the identity provider administrator that you configured single logout for the Fedlet service provider, so that any additional required changes can be made to the identity provider configuration.

.NET Fedlet Service Provider Initiated Single Sign-on (CR 6928525)

The .NET Fedlet supports the SAMLv2 service provider initiated single sign-on (SSO). In addition, artifact support is required to allow the .NET Fedlet to receive an artifact and then have it resolved through SOAP with the issuing identity provider's Artifact Resolution Service.

The .NET Fedlet sample application shows how you can configure single sign-on. After your application has the necessary artifacts installed, a specific URI is required to receive the HTTP POST containing the SAMLv2 response after successful authentication by the identity provider. The following code example shows how you can retrieve this information in a .NET application:

EXAMPLE 5–2 Code Example to Retrieve the `AuthnResponse` in a .NET Fedlet Application

```
AuthnResponse authnResponse = null;
try
{
      ServiceProviderUtility spu = new ServiceProviderUtility(Context);
      authnResponse = spu.GetAuthnResponse(Context);
}
catch (Saml2Exception se)
{
      // invalid AuthnResponse received
}
catch (ServiceProviderUtilityException spue)
{
      // issues with deployment (reading metadata)
}
```

If your application receives the SAMLv2 response, the `authnResponse` object will be populated with the assertion information. The sample application shows how to retrieve the attributes and subject information from this object.

.NET Fedlet Support for Multiple Identity Providers and Discovery Service (CR 6928524)

The .NET Fedlet supports multiple identity providers and the identity provider discovery service.

In some deployments, you might want to configure the .NET Fedlet with multiple identity providers such as Oracle OpenSSO 8.0 Update 2. Perform the following task for each additional identity provider you wan to add.

▼ To Configure the .NET Fedlet for Multiple Identity Providers

1. **Get the XML metadata file from the additional identity provider.**

2. **Name the additional identity provider metadata file as `idp`*n*`.xml`, where *n* is the identity provider that you are adding. For example, name the second identity provider file as `idp2.xml`, the third as `idp3.xml`, and so on. This procedure uses `idp2.xml` as the file name.**

3. **Copy the `idp2.xml` file from Step 2 to your application's `App_Data` folder.**

4. **Add this new identity provider to the .NET Fedlet circle of trust.**

   To add the new identity provider to an existing circle of trust:

   In the `fedlet.cot` file in your application's `App_Data` folder, append the new IDP entity ID (indicated by the `entityID` attribute in the `idp2.xml` metadata file) to the value of the `sun-fm-trusted-providers` attribute, using a comma (,) as a separator.

   To add the new identity provider to a new circle of trust:

   a. **Create a new file named `fedlet2.cot` in your application's `App_Data` folder. Use the existing `fedlet.cot` as a template, but change the value of the `cot-name` attribute to the name of the new circle of trust (for example, `cot2`). Include both the new identity provider entity ID and the Fedlet entity ID as value for the `sun-fm-trusted-providers` attribute, with the two entity IDs separated by a comma (,).**

   b. **In the `sp-extended.xml` file, add the new circle of trust name to the value of the `cotlist` attribute. For example, for a circle of trust named `cot2`:**

      ```
      <Attribute name="cotlist">
      <Value>saml2cot</Value>
      ```

```
<Value>cot2</Value>
</Attribute>
```

5. **In your application's `App_Data` folder, create a new `idp2-extended.xml` file as the extended metadata for the new identity provider. Use the existing `idp-extended.xml` file as a template, but change the `entityID` to the new identity provider entity ID. Change the value for the `cotlist` attribute to the circle of trust name, if a new circle of trust is created for the identity provider. Make sure that the additional identity provider is a remote identity.**

6. **Restart the Application Pool associated with your Fedlet .NET application.**

7. **The Fedlet metadata XML file (`sp.xml`) must be imported into the additional identity provider and added to the same circle of trust as the identity provider entity. Either import the `sp.xml` file into the identity provider, or give the file to your identity provider administrator to import.**

.NET Fedlet Support for the Identity Provider Discovery Service (CR 6928524)

In this scenario, the .NET Fedlet is configured with multiple identity providers in a circle of trust and you want to configure the Fedlet to use the identity provider discovery service to determine the preferred identity provider.

The discovery service must be configured for the identity providers you are using with the .NET Fedlet. For information about configuring the identity provider discovery service in Oracle OpenSSO 8.0 Update 2, see the following documentation collection:

`http://download.oracle.com/docs/cd/E19681-01/index.html`

▼ To Configure the .NET Fedlet to Use the Identity Provider Discovery Service:

1. **In the .NET Fedlet `fedlet.cot` file, set the `sun-fm-saml2-readerservice-url` property to the SAMLv2 reader service URL. For example:**

   ```
   sun-fm-saml2-readerservice-url=http://discovery.common.com/opensso/saml2reader
   ```

2. **Restart the Application Pool associated with your .NET Fedlet application.**

Documentation Errata

The Fedlet Java API reference is available in the Oracle OpenSSO 8.0 Update 2 Java API Reference in the following documentation collection:

`http://download.oracle.com/docs/cd/E19681-01/index.html`

Note – The `getPolicyDecisionForFedlet` method is not supported in the OpenSSO 8.0 Update 2 release.

◆ ◆ ◆ CHAPTER 6

Integrating the OpenSSO 8.0 Update 2 with Oracle Access Manager

This chapter provides instructions for implementing single sign-on using OpenSSO 8.0 Update 2 and Oracle Access Manager 10g or 11g. This information supplements conceptual information contained in Chapter 3, "Integrating Oracle Access Manager," in *Sun OpenSSO Enterprise 8.0 Integration Guide*. This use case provides a single sign-on experience to OpenSSO-protected applications by honoring an Oracle Access Manager session. The configured OpenSSO authentication module generates an OpenSSO session based on the Oracle Access Manager session.

Overview of Integration Steps

Before You Begin

Be sure you have access to the following components before you attempt to install OpenSSO 8.0 Update 2 for integration with Oracle Access Manager:

opensso.zip	This zip file contains the opensso.war file, integration source code, configuration files and other tools that are required for OpenSSO 8.0 Update 2 installation and configuration.

OpenSSO Agent	The OpenSSO Agent is used when an application protected by OpenSSO can actually use the authentication session established by Oracle Access Manager.
Oracle Access Manager 10g or 11g	Download Oracle Access Manager from Oracle web site. See the `http://www.oracle.com/technetwork/middleware/weblogic/downloads/index.html` page.
Oracle Web Gate 10g or 11g	Download Oracle Webgate for a container that is supported by both OpenSSO and Oracle Webgate. At this time, Sun Web Server 7.x is the only container that is supported by both the products. See the `http://www.oracle.com/technetwork/middleware/weblogic/downloads/index.html`page.
Oracle Access Manager SDK 10g or 11g	Download Oracle Access Manager. The SDK is required to compile and build OpenSSO Authentication Modules for Oracle Access Manager integration. See the See the `http://www.oracle.com/technetwork/middleware/weblogic/downloads/index.html` page
OpenSSO C-SDK 2.2	(Optional) The OpenSSO C-SDK is required for creating an authentication module in Oracle Access Manager itself to generate an OAM session. This may not be a common use case from OpenSSO perspective. See "Where is the C SDK?" in *Sun OpenSSO Enterprise 8.0 C API Reference for Application and Web Policy Agent Developers*

Unpacking the Integration Bits

The `opensso/integrations/oracle` directory contains source and configurations to compile and build custom authentication modules and other plugins. See Chapter 3, "Integrating Oracle Access Manager," in *Sun OpenSSO Enterprise 8.0 Integration Guide* for use case options and related information. The following table summarizes the files under `opensso/integrations/oracle` directory and descriptions for each file.

`README.html`	This is the file you're reading now.

`build.xml`	An ant build file for building a custom authentication module for Oracle Access Manager in OpenSSO
`config`	Configuration files required for creating an authentication module for Oracle Access Manager in OpenSSO. ■ `OblixAuthService.xml` Authentication service file for Oracle Access Manager authentication module ■ `OblixAuthModule.xml` Authentication module callbacks for Oracle Access Manager. This is an empty file by default, but it must be present for configuration purposes. ■ `OblixAuth.properties` Properties file that stores internationalization keys for the authentication
`lib`	This directory is empty by default. This `lib` directory must contain the following libraries to compile the source libraries. ■ `jobaccess.jar` Copy this file from the Oracle Access Manager SDK. ■ `openfedlib.jar`, `amserver.jar`, and `opensso-sharedlib.jar` Copy these files from opensso.war ■ `servlet.jar or javaee.jar` Copy the GlassFish `lib` directory. Ideally, any JAR file that has standard Java EE classes such as `javax.servlet.http.Cookie` is fine.
`source`	Directory containing the following source files: ■ `com/sun/identity/authentication/oblix/OblixAuthModule.java` ■ `com/sun/identity/authentication/oblix/OblixAuthModule.java` ■ `com/sun/identity/authentication/oblix/OblixPrincipal.java` ■ `com/sun/identity/saml2/plugins/OAMAdapter.java`

This class is a SAML2 Plugin Adapter for SAML Service Providers. This class does the remote authentication to Oracle Access Manager using the OpenSSO Session service.

oamauth (optional) This directory contains source files for Oblix Authentication Scheme for OpenSSO. This is a C-based authentication module and leverages the OpenSSO C-SDK for validation.

- oam/solaris/authn_api.c

  This file implements Oblix custom authentication scheme for OpenSSO.

- oam/solaris/include/*.h

  All the header files that are required to compile auth scheme.

- oam/solaris/AMAgent.properties

  Sample OpenSSO Agent configuration file. This is required for the authentication scheme to validate the OpenSSO session.

Building Source Files for Oracle Access Manager in OpenSSO

Use the ant script to build the source files. A compatible ant script must be installed and configured in the PATH.

▼ To Build the Source Files for Oracle Access Manager

1 Run the following command:

```
cd $openssozipdir/integrations/oracle; ant -f build.xml
```

This command builds source files and generates `fam_oam_integration.jar` into the `$openssozipdir/integrations/oracle/dist` directory.

2 Bundle the authentication module into the OpenSSO WAR file.

a. Create a temporary directory and unwar the `opensso.war`. Example:

```
# mkdir /export/tmp
# cd /export/tmp
# jar -xvf opensso.war
```

From now on, `/export/tmp` is used as a WAR staging area, and is represented with a marco `$WAR_DIR`.

b. Copy `$openssozipdir/integrations/oracle/dist/fam_oam_integration.jar` to `$WAR_DIR/WEB-INF/lib`.

c. Copy `$openssozipdir/integrations/oracle/config/OblixAuth.properties` to `$WAR_DIR/WEB-INF/classes`.

d. Copy `$openssozipdir/integrations/oracle/config/OblixAuthModule.xml` to `$WAR_DIR/config/auth/default`, and also to the directory `$WAR_DIR/config/auth/default_en`.

e. Re-war `opensso.war` using `jar cvf opensso.war` from `$WAR_DIR`.

(Optional) Build an Authentication Scheme for OpenSSO in Oracle Access Manager

Note: This is not a common use case. You do not have to build this unless it is required, such as in a SAML2 service provider use case.

To build the Oblix authentication scheme, you must customize the `makefile`. Also, since this is a C-based authentication module, it is operating system-dependent.

▼ To Build an Authentication Scheme for OpenSSO in Oracle Access Manager

Before You Begin The authentication scheme files are located under the `$openssozipdir/integrations/oracle/oamauth/solaris` directory.

1 **Download and configure the OpenSSO C-SDK 2.2 version.**

The `authn_api.c` file contains a reference to `AMAgent.propeties file`. Modify the file accordingly.

2 **Customize `makefile` for your environment.**

For example, specify the `gcc` compile location. Also edit the `LDFLAGS` to point to your OpenSSO C-SDK `lib` directory.

3 **Run the `make` command.**

The `make` command should result in an `authn_api.so` file.

Configuring Single Sign-On Using Oracle Access Manager and Oracle OpenSSO STS

▼ To Configure Single Sign-On Using Oracle Access Manager and Oracle OpenSSO 8.0 Update 2

Before you begin: Sun Java System Web Server 7.x must already be installed and configured. See the Sun Java System Web Server Documentation Wiki for Web Server installation instructions.

1. **Install OpenSSO on Sun Java System Web Server 7.x.**

2. **Install an OpenSSO Policy Agent on a supported container and configure the agent to work with OpenSSO.**

   For installation instructions, see the Policy Agent 3.0 guide for the agent you are using. These guides are available in the following documentation collection:

   `http://download.oracle.com/docs/cd/E19681-01/index.html`

3. **Install and configure Oracle Access Manager.**

   See the *Oracle Access Manager Installation Guide 10g (10.1.4.3)*

4. **Install and configure Oracle Access Manager SDK with Oracle Access Manager.**

   See the *Oracle Access Manager Installation Guide 10g (10.1.4.3)*

5. **Install Oracle Webgate on the same web container where OpenSSO server is installed. (Sun Web Server 7.x)**

   Configure OpenSSO so that it protects only `deployURI/UI/*` of the OpenSSO web application. Example:`/opensso/UI/.../*`

   For Oracle Access Manager policies, resources and other configuration details, check the Oracle Access Manager administration guide. Unprotect every other URL in OpenSSO Enteprise. This is for simple single sign-on integration scenario, but evaluate policies based on full integration and other deployment dependencies.

6 Configure the Authentication Module in OpenSSO.

a. Access the OpenSSO console.

The browser redirects to Oracle Access Manager for authentication. After successful authentication, OpenSSO presents a login page. Log in using the OpenSSO admin user name and password.

b. Import the Oracle Authentication Module service XML file into the OpenSSO configuration.

The authentication module service can be loaded from command line `ssoadm` utility, and as well as browser based ssoadm.jsp.

c. Access `http://host:port/opensso/ssoadm.jsp`.

d. Choose the create-service option.

e. Copy and paste the XML file from `$openssozipdir/integrations/oracle/config/OblixAuthService.xml` and click Submit.

This loads the authentication module service into the OpenSSO configuration.

f. Register the authentication module into the authentication Core service.

The Core service contains a list of authenticators. Choose the `register-auth-module` option in `http://host:port/opensso/ssoadm.jsp`. Enter `com.sun.identity.authentication.oblix.OblixAuthModule` as the authentication module class name.

g. Verify that the authentication module is registered to the default realm.

Access OpenSSO using the URL http://host:port/opensso. In the OpenSSO console, click the default realm, and then click the Authentication tab. Click New to create a new authentication module named OblixAuth.

h. On the Authentication tab, select the OblixAuth authentication module.

Configure the Oblix SDK directory. Enable Check Remote User Header Only, and specify the remote header name as `OAM_REMOTE_USER`. This parameter is configurable based on the deployment.

7 (Optional) Enable the Ignore Profile option in the OpenSSO core authentication service.

In the OpenSSO console, go to Configuration > Core > Realm Attributes > User Profile . Choose Ignored, and then click Save.

This configuration prevents OpenSSO from searching for an existing user profile after successful authentication. However, if the user repository used by OpenSSO and Oracle Access

Manager are exactly same, then this step is not necessary. Go to Admin Console -> Configuration -> Core -> Realm Attributes -> User Profile. Choose Ignored, and then click Save.

8 **Edit the web server start script to include Oracle Access Manager SDK shared libraries.**

Update `LD_LIBRARY_PATH` in the `startserv` script to include the shared libraries from `$ACCESSDKDIR/oblix/lib`.

9 **Restart the Sun Web Server that contains both OpenSSO and Oracle Webgate.**

10 **Update the Login URL for Web Agent value as `http://openssohost:openssoport/deployURI/UI/Login?module=OblixAuth`.**

To Test Single Sign-On

Access the protected resource from the OpenSSO-protected application. The browser should redirect you to the Oracle Access Manager Login Page if you are not already authenticated. After successful login, it creates an OpenSSO session, and finally redirects back to the Policy Agent-protected application URL . Based on the policy, you are allowed or denied access to the protected application.

(Optional) Installing of Oblix AuthScheme into Oracle Access Manager

This is useful when the Oracle Access Manager session must be generated upon validating the OpenSSO session. See Chapter 3, "Integrating Oracle Access Manager," in *Sun OpenSSO Enterprise 8.0 Integration Guide*for information about relevant use cases.

The Oblix Authentication Schemes are exposed as C authentication modules, and this authentication scheme uses OpenSSO C-SDK 2.2 version to validate the OpenSSO Session. The OpenSSO Authentication Scheme in Oblix uses a configuration for the OpenSSO client-side configuration in `AMAgent.properties`. This file must be customized before configuring the authentication module. The build instructions specify the location of this file. The compiled `authn_api.so` and other C-SDK libraries must be copied to the `$OAM_INSTALL_DIR/access/oblix/lib` directory before configuring the Authentication Scheme. The *Sun OpenSSO 8.0 Integration Guide* shows a sample screen shot illustrating how to configure the Oracle Authentication Scheme, and this should be used as a reference only. For more details, see the latest Oracle Access Manager documentation.

Integrating the OpenSSO 8.0 Update 2 with Oracle Access Manager

This section provides instructions for implementing single sign-on using OpenSSO 8.0 Update 2 and Oracle Access Manager versions 10.1.4.0.1. and 11g. This information supplements conceptual information contained in Chapter 3, "Integrating Oracle Access Manager," in *Sun OpenSSO Enterprise 8.0 Integration Guide*. This use case provides a single sign-on experience to OpenSSO-protected applications by honoring an Oracle Access Manager session. The configured OpenSSO authentication module generates an OpenSSO session based on the Oracle Access Manager session.