faq

ORACLE FUSION MIDDLEWARE
Oracle Application Server 10*g*
Platform Security
Frequently Asked Questions
June 2006

Introduction

This FAQ addresses frequently asked questions relating to the security features of Oracle Application Server (Oracle AS) 10g Release 3. This FAQ is broken into the following sections:

1.0 Single Sign-On

1.1 What is OracleAS Single Sign-On (SSO)?

OracleAS SSO supports single sign-on for web (browser) clients. OracleAS SSO allows web users who access OracleAS to sign in once, and be authenticated to multiple web applications, including Oracle Portal, Oracle E-Business Suite and non-Oracle applications. These web applications are classified as either partner or external applications.

Partner applications are those that work within the OracleAS SSO framework. They are designed, or have been modified to delegate responsibility for user authentication to OracleAS SSO. They accept the user identity presented to them by OracleAS SSO. Since partner applications rely on the authentication services of OracleAS SSO, they do not need to implement their own authentication modules. User administration is simplified for partner applications, since there is no need to manage identities or passwords for these applications. Deploying an application as a partner application can thus reduce both development and ongoing administrative costs.

External applications are those that retain their own usernames and passwords, and do not delegate responsibility for authenticating users to OracleAS SSO. These applications have not been developed or modified to work within the SSO framework. A typical external application might be one developed or deployed by a third party, such as a portal website which requires username and password for access to custom services like email. Although partner applications are preferable, external applications allow existing or legacy applications to work with OracleAS SSO without any retrofitting.

1.2 What are the key strengths of OracleAS SSO?

OracleAS SSO's main strengths are its flexibility, standards-based approach, and scalable design. These are described below:

- Flexibility: OracleAS SSO is designed to work standalone or with your existing infrastructure, including other SSO servers like Oracle Access Manager, and LDAP directories.
- Standard-based approach: OracleAS SSO uses Internet standards including HTTP(s) for communication, cookies, and X.509 certificates for user tokens.
- Scalable design: OracleAS SSO is designed to support terabytes of user information persisted within Oracle Internet Directory (OID).

1.4 What are the main new SSO features in OracleAS 10g?

The main new features in this release include:

- Multilevel authentication: Brief description required
- Windows Native Authentication: Brief description required
- Flexible deployment options: Brief description required

1.5 How does Oracle SSO affect overall user experience versus a non-SSO environment?

For the end user, SSO saves time by requiring only a single sign-on. After the first login to an application, the user no longer sees login screens for any SSO enabled applications, unless there are timeouts or deliberate security settings that force additional checks. The end result for the user is less keystrokes and reduced steps to access applications.

1.6 How does SSO work with Java and J2EE applications?

From the OracleAS SSO perspective, Java applications are no different than any other application. Java developers are encouraged, however, to take advantage of the OracleAS JAAS Provider. JAAS is a specification from the Java community that provides simple APIs for authentication and authorization. Oracle's JAAS Provider is integrated with the OracleAS SSO, giving Java developers a seamless way to support single sign-on.

1.6 How does one verify that SSO is working?

In the $ORACLE_HOME of the mod_osso you want to test, create a private directory and an index.html file

1. Navigate to the $ORACLE_HOME/Apache/Apache/htdocs
2. Create a directory called "private"

   Unix: $ORACLE_HOME/Apache/Apache/htdocs/private
   Windows: %ORACLE_HOME%\Apache\Apache\htdocs\private

3. Browse to that directory, for example: cd private
4. Create an index.html file in the directory:

Unix: $ORACLE_HOME/Apache/Apache/htdocs/private/index.html
Windows: %ORACLE_HOME%\Apache\Apache\htdocs\private\index.html

5. Test accessibility via a web browser. From a browser Access the private directory URL:

http://ssoservername.domain:port/private/index.html

This should display the static index.html page. If it does, then go to step 6. If it does not, you will need to resolve your http server issue.

6. Backup mod_osso.conf before modifying it. Edit the mod_osso.conf file in order to protect the "/private" directory.

   Unix: $ORACLE_HOME/Apache/Apache/conf/mod_osso.conf
   Windows: %ORACLE_HOME%\Apache\Apache\conf\mod_osso.conf

```
----Before-----
#  <Location /private>
#   AuthType Basic
# Require valid-user
#  </Location>

------After------
  <Location /private>
   AuthType Basic
 Require valid-user
  </Location>

Example:
  <IfModule mod_osso.c>
     OssoIpCheck on
     OssoIdleTimeout off
     OssoConfigFile /home/product/oracleas/Apache/Apache/conf/osso/osso.conf
    <Location /private>
       AuthType Basic
       Require valid-user
    </Location>
  </IfModule>
```

NOTE: If you edited the mod_osso.conf file manually, in 10.1.2 you would need to issue dcmctl command as follows:

```
dcmctl updateConfig -v -d
```

7. Restart the associated http server either from EM website or from the command prompt.

```
opmnctl stopproc process-type=HTTP_Server

opmnctl startproc process-type=HTTP_Server
```

8. Test accessibility via a web browser. From a browser Access the private directory URL:

   http://ssoservername.domain:port/private/index.html

This should redirect you to the SSO login prompt.

9. Login using the 'orcladmin' user and the password for the ias_admin user (by default).
This should display the text from the static index.html file created in step 4
If the index.html is displayed, then mod_osso is working. If it is not, then there is a core issue with mod_osso.

2.0 OracleAS JAAS Provider

JAAS is part of the standard Java2 Security Model, which it extends by defining a standard pluggable Authentication Module (PAM) framework. The provision of a standard framework allows the definition of application security requirements to be fully divorced from the mechanisms that can be used to authenticate their identity.

2.1 What is the OracleAS JAAS Provider?

JAAS stands for Java Authentication and Authorization Services. Oracle's JAAS implementation, known officially as the OracleAS JAAS Provider, provides core security services for developing Java-based applications for OracleAS. Oracle's JAAS Provider is part of a larger set of security services in OracleAS that includes single sign-on, network encryption and other features.

2.2 How does Oracle's JAAS Provider fit into the Java security model?

JAAS as a specification is not yet integrated with the Java 2 Platform, Enterprise Edition (J2EE) security model. However, Oracle's JAAS Provider does provide security for Oracle Containers for Java (OC4J), Oracle's J2EE implementation, to enforce security constraints for Web Servlets, Java Server Pages (JSPs) and EJB components.

2.3 What are the security services provided by Oracle's JAAS Provider?

Oracle's JAAS Provider provides key security services for:

- Authentication: identifying users
- Authorization: controlling what they can do
- Delegation: Enabling code to run securely, with the privileges of other users.

2.4 What are the key benefits of Oracle's JAAS Provider?

Oracle's JAAS Provider provides benefits to any customer developing Java-based applications. One of the biggest benefits of Oracle's JAAS Provider is integration with OracleAS Single Sign-On (SSO). This integration enables any Java-based application to participate in web single sign-on.

2.5 What are the JAZN provider types and which types are provided for OracleAS?

Oracle's JAAS implementation offers two JAAS provider types

- OracleAS JAAS XML: This is a lightweight version where user credentials and privileges are defined in an XML document.
- OracleAS JAAS LDAP: This is the enterprise ready version, where user credentials and privileges are retrieved from an LDAP store. Sets of user credentials can be organized by assigning them to a realm.

2.6 What is a realm?

The XML file used by JAZN-XML and the Oracle Internet Directory (OID) repository used by JAZN-LDAP can be used to represent multiple, distinct sets of user credentials and privileges. Each distinct set of user credentials and privileges is known as a realm.

2.7 What is the default realm?

A default realm is created during the installation of Application Server. During the post installation phase, configuration assistants for OC4J create the default configuration for the JAZN-LDAP provider and, in the case of the Oracle Application Server "Infrastructure" install type, create the required DIT structure in OID

2.8 How and where is the information about user credentials and privileges associated with a realm represented?

Each provider type represents information differently. In the case of JAZN-XML, each REALM is defined as sets of XML entries nested within a <realm> element and so can easily be determined by inspecting the XML file using an editor. For example:

```
<realm>
  <name>jazn.com</name>
  <users>
    <user>
      <name>SCOTT</name>
      <description>SCOTT</description>
      <credentials>{903}iDn3ZdvhVUAPpD1Kqy78hlUlyrMIgyj</credentials>
    </user>
  </users>
  <roles>
    <role>
      <name>users</name>
      <members>
        <member>
          <type>user</type>
          <name>SCOTT</name>
        </member>
      </members>
    </role>
  </roles>
</realm>
```

In this case, the default realm is "jazn.com". In the case of JAZN-LDAP, the information is represented as branches of the Directory Information Tree (DIT) representing hierarchies of specific LDAP object types and expected relationships between entries.

3.0 OracleAS SSL

2.7 How do I configure the OracleAS infrastructure to use SSL?

Oracle provides a tool named SSLConfigTool to configure the OracleAS 10g release 3 infrastructure install with SSL. The following is an example of running this tool to configure SSL.

1. Create the $ORACLE_HOME/ssl_config directory. NOTE: SSLConfigTool creates log and ldif files when it is run, therefore it is prudent to create a directory to where these files will be written.
2. Navigate to the $ORACLE_HOME/ssl_config directory
3. Run the SSL Configuration Tool (located in $ORACLE_HOME/bin)

   SSLConfigTool -config_w_default -secure_admin -opwd <orcladmin password>

The tool is meant to be run only once from a given Oracle Home. If for any reason, you need to run it a second time, you must revert the initial changes first by running the tool and specifying the rollback parameter: *SSLConfigTool* -rollback -*opwd <orcladmin password>.*

At installation, Oracle Internet Directory starts up with configset0, which specifies dual mode. That is, some components can access Oracle Internet Directory using non-SSL connections, while others use SSL when connecting to the directory. By default, Oracle Application Server components are configured to run in this dual mode environment when communicating with OID. If preferred,,you can remove the non-SSL mode and change all middle-tier instances to use only SSL. For more information, please refer to the section on changing OID from dual mode to SSL mode in the Oracle Application Server Administrator's Guide.

ORACLE FUSION MIDDLEWARE

Oracle Application Server 10*g*: <Component Name> FAQ

Mon DD, YYYY
Author:
Contributing Authors:

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Oracle, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.