EPM System Security Administration Guide, 11.1.2.2

Copyright © 2005,2014, Oracle and/or its affiliates. All rights reserved.

Authors: EPM Information Development Team

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

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

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

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

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

U.S. GOVERNMENT END USERS:

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

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

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Documentation Accessibility</strong></td>
<td>7</td>
</tr>
<tr>
<td><strong>Documentation Feedback</strong></td>
<td>9</td>
</tr>
<tr>
<td><strong>Chapter 1. About EPM System Security</strong></td>
<td>11</td>
</tr>
<tr>
<td>About EPM System Products</td>
<td>11</td>
</tr>
<tr>
<td>Assumed Knowledge</td>
<td>11</td>
</tr>
<tr>
<td>Security Infrastructure Components</td>
<td>12</td>
</tr>
<tr>
<td>User Authentication</td>
<td>12</td>
</tr>
<tr>
<td>Authentication Components</td>
<td>12</td>
</tr>
<tr>
<td>Default EPM System Single Sign-on</td>
<td>13</td>
</tr>
<tr>
<td>Single Sign-on from Access Management Systems</td>
<td>14</td>
</tr>
<tr>
<td>Provisioning (Role-Based Authorization)</td>
<td>16</td>
</tr>
<tr>
<td>Roles</td>
<td>17</td>
</tr>
<tr>
<td>Users</td>
<td>18</td>
</tr>
<tr>
<td>Groups</td>
<td>18</td>
</tr>
<tr>
<td><strong>Chapter 2. SSL-Enabling EPM System Components</strong></td>
<td>19</td>
</tr>
<tr>
<td>Assumptions</td>
<td>19</td>
</tr>
<tr>
<td>Information Sources</td>
<td>19</td>
</tr>
<tr>
<td>Location References</td>
<td>20</td>
</tr>
<tr>
<td>About SSL-Enabling EPM System Products</td>
<td>20</td>
</tr>
<tr>
<td>Supported SSL Scenarios</td>
<td>21</td>
</tr>
<tr>
<td>Required Certificates</td>
<td>21</td>
</tr>
<tr>
<td>Terminating SSL at the SSL Offloader</td>
<td>22</td>
</tr>
<tr>
<td>Deployment Architecture</td>
<td>22</td>
</tr>
<tr>
<td>Assumptions</td>
<td>23</td>
</tr>
<tr>
<td>Configuring EPM System</td>
<td>24</td>
</tr>
<tr>
<td>Testing the Deployment</td>
<td>24</td>
</tr>
<tr>
<td>Full SSL Deployment of EPM System</td>
<td>24</td>
</tr>
<tr>
<td>Deployment Architecture</td>
<td>24</td>
</tr>
<tr>
<td>Assumptions</td>
<td>25</td>
</tr>
<tr>
<td>Configuring EPM System for Full SSL</td>
<td>27</td>
</tr>
</tbody>
</table>
Appendix B. Implementing a Custom Login Class ............................................ 99
  Custom Login Class Sample Code ......................................................... 99
  Deploying a Custom Login Class ......................................................... 102

Appendix C. Using the Update Native Directory Utility ........................................ 103
  About the Update Native Directory Utility ........................................... 103
  Update Native Directory Utility Installation Location ............................. 103
  Update Native Directory Utility Options .............................................. 103
  Using Update Native Directory Utility .................................................. 104
    Updating the Update Native Directory Utility Settings .......................... 104
    Identifying Stale Data ................................................................. 104
    Deleting Stale Data .................................................................. 105
  Log Files Generated by Update Native Directory Utility ........................... 106

Appendix D. Migrating Users and Groups Across User Directories ......................... 107
  Overview .......................................................................................... 107
  Prerequisites ......................................................................................... 107
  Migration Procedure ........................................................................... 108
    Export Native Directory Data .......................................................... 108
    Prepare EPM System for Migration ................................................... 109
    Restart EPM System ................................................................... 110
    Edit Import Files ........................................................................ 110
    Import Updated Data .................................................................. 110
    Run Update Native Directory Utility .............................................. 111
  Product-Specific Updates .................................................................... 111
    Planning ......................................................................................... 111
    Financial Management ................................................................. 111
    Reporting and Analysis ............................................................... 112

Glossary .................................................................................................. 113
Documentation Accessibility

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

Access to Oracle Support

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

Send feedback on this documentation to: epmdoc ww@oracle.com

Follow EPM Information Development on these social media sites:
LinkedIn - http://www.linkedin.com/groups?gid=3127051&goback=.gmp_3127051
Twitter - http://twitter.com/hyperioneppinfo
Facebook - http://www.facebook.com/pages/Hyperion-EPM-Info/102682103112642
Google+ - https://plus.google.com/106915048672979407731/#106915048672979407731/posts
YouTube - http://www.youtube.com/user/OracleEPMWebcasts
About EPM System Products

Oracle Enterprise Performance Management System products form a comprehensive enterprisewide system that integrates modular suites of financial management and planning applications with the most comprehensive business intelligence capabilities for reporting and analysis. Major components of EPM System products:

- Oracle Hyperion Foundation Services
- Oracle Essbase
- Oracle Hyperion Financial Management
- Oracle Hyperion Planning
- Oracle Hyperion Reporting and Analysis

For information about the products and components in each of these product families, see Oracle Enterprise Performance Management System Installation Start Here.

Assumed Knowledge

This guide is for administrators who configure, secure, and manage EPM System products. It assumes the following knowledge:

- A strong understanding of your organization’s security infrastructure, including the following:
  - Directory servers; for example, Oracle Internet Directory, Sun Java System Directory Server, and Microsoft Active Directory
  - Use of Secure Socket Layer (SSL) to secure communication channels
Security Infrastructure Components

EPM System integrates a number of security components to ensure robust application security. When integrated into a secure infrastructure, EPM System delivers a highly secure suite of applications that ensures data and access security. The infrastructure components that you can use to secure EPM System include:

- An optional access management system; for example, Oracle Access Manager to provide SSO access to EPM System products
- Use of an integrated SSO infrastructure; for example, Kerberos.
  You can use Kerberos authentication with the access management system (SiteMinder) to ensure that Windows users can transparently log into SiteMinder and EPM System products.
- Use of Secure Socket Layer (SSL) to secure communication channels among EPM System products and clients

User Authentication

User authentication enables single sign-on (SSO) functionality across EPM System products by validating the login information of each user to determine authenticated users. User authentication, along with product-specific authorization, grants the user access to EPM System products. The process of granting authorization is called provisioning.

Authentication Components

The following sections describe the components that support SSO:

- “Native Directory” on page 12
- “External User Directories” on page 13

Native Directory

Native Directory refers to the relational database that Oracle Hyperion Shared Services uses to support provisioning and to store seed data such as default user accounts.

Native Directory functions:

- Maintain and manage the default EPM System user accounts
- Store all EPM System provisioning information (relationships among users, groups, and roles)

**External User Directories**

User directories refer to corporate user and identity management systems that are compatible with EPM System products.

EPM System products are supported on several user directories, including LDAP-based user directories; for example, Oracle Internet Directory, Sun Java System Directory Server (formerly SunONE Directory Server), and Microsoft Active Directory. Relational databases also are supported as user directories. User directories other than Native Directory are referred to as external user directories throughout this document. See Oracle Enterprise Performance Management Products—Supported Platforms Matrices for a list of supported user directories.

From Shared Services Console, you can configure many external user directories as the source for EPM System users and groups. Each EPM System user must have a unique account in one of the configured user directories. Generally, EPM System users are assigned to groups to facilitate provisioning.

**Default EPM System Single Sign-on**

EPM System support SSO across EPM System web applications by allowing authenticated users from an application to seamlessly navigate to other applications without reentering credentials. SSO is implemented by integrating a common security environment that handles user authentication and provisioning (role-based authorization) across EPM System products.

The default SSO process is depicted in the following illustration.
1. Through a browser, users access an EPM System product login screen and enter user names and passwords. The EPM System product queries the configured user directories (including Native Directory) to verify user credentials. Upon finding a matching user account in a user directory, the search is terminated, and the user’s information is returned to the EPM System product. Access is denied if a user account is not found in any configured user directory.

2. Using the retrieved user information, the EPM System product queries Native Directory to obtain provisioning details for the user.

3. EPM System product checks the Access Control List (ACL) in the product to determine the application artifacts that the user can access.

Upon receiving provisioning information from Native Directory, the EPM System product is made available to the user. At this point, SSO is enabled for all EPM System products for which the user is provisioned.

**Single Sign-on from Access Management Systems**

To further secure EPM System products, you can implement a supported access management system such as Oracle Access Manager or SiteMinder, which can provide authenticated user credentials to EPM System products and control access based on predefined access privileges.

SSO from security agents is available for EPM System web applications only. In this scenario, EPM System products use the user information provided by the security agent to determine access permissions of users. To enhance security, Oracle recommends that direct access to the servers be blocked by firewalls so that all requests are routed through an SSO portal.
SSO from access management systems is supported by accepting authenticated user credentials through an acceptable SSO mechanism. See “Supported SSO Methods” on page 47. The access management system authenticates users and passes their login name to EPM System. EPM System verifies the login names against configured user directories.

See these topics.

- “Single Sign-on from Oracle Access Manager” on page 49
- “OracleAS Single Sign-on” on page 51
- “SiteMinder SSO” on page 61
- “Kerberos Single Sign-on” on page 64

The illustrated concept:

1. Using a browser, users request access to a resource protected by an access management system, for example; Oracle Access Manager, or SiteMinder.

   **Note:** EPM System products are defined as resources protected by the access management system.

   The access management system intercepts the request and presents a login screen. Users enter user names and passwords, which are validated against configured user directories in the access management system to verify user authenticity. EPM System products are also configured to work with these user directories.

   Information about the authenticated user is passed to the EPM System product, which accepts the information as valid.
The access management system passes the user’s login name (value of Login Attribute) to the EPM System product using an acceptable SSO mechanism. See “Supported SSO Methods” on page 47.

2. To verify user credentials, the EPM System product tries to locate the user in a user directory. If a matching user account is found, user information is returned to the EPM System product. EPM System security sets the SSO token that enables SSO across EPM System products.

3. Using the retrieved user information, the EPM System product queries the Native Directory to obtain provisioning details for the user.

   Upon receiving user provisioning information, the EPM System product is made available to the user. SSO is enabled for all EPM System products for which the user is provisioned.

**Provisioning (Role-Based Authorization)**

EPM System security determines user access to applications using the concept of roles. Roles are permissions that determine user access to application functions. Some EPM System products enforce object-level ACLs to further refine user access to their artifacts, such as reports and members.

Each EPM System product provides several default roles tailored to various business needs. Each application belonging to an EPM System product inherits these roles. Predefined roles from the applications registered with Shared Services are available from Shared Services Console. You may also create additional roles that aggregate the default roles to suit specific requirements. These roles are used for provisioning. The process of granting users and groups specific roles belonging to EPM System applications and their resources is called **provisioning**.

Native Directory and configured user directories are sources for user and group information for the provisioning process. You can browse and provision users and groups from all configured user directories from Shared Services Console. You can also use application-specific aggregated roles created in Native Directory in the provisioning process.

An illustrated overview of the authorization process:
1. After a user is authenticated, EPM System product queries user directories to determine the user’s groups.

2. The EPM System product uses group and user information to retrieve the user’s provisioning data from Shared Services. The product uses this data to determine which resources a user can access.

   Product-specific provisioning tasks, such as setting product-specific access control, are completed for each product. This data is combined with provisioning data to determine the product access for users.

Role-based provisioning of EPM System products uses these concepts.

### Roles

A role is a construct (similar to an access control list) that defines the access permissions granted to users and groups to perform functions on EPM System resources. It is a combination of resource or resource types (what users can access, for example, a report) and actions that users can perform on the resource (for example, view and edit).

Access to EPM System application resources is restricted. Users can access them only after a role that provides access is assigned to the user or to the group to which the user belongs. Access restrictions based on roles enable administrators to control and manage application access.

### Global Roles

Global roles, which are Shared Services roles that span multiple products, enable users to perform certain tasks across EPM System products. For example, the Shared Services Administrator can provision users for all EPM System applications.
Predefined Roles

Predefined roles are built-in roles in EPM System products. You cannot delete them. Each application instance belonging to an EPM System product inherits the predefined roles of the product. These roles, for each application, are registered with Shared Services when you create the application.

Aggregated Roles

Aggregated roles, also known as custom roles, aggregate multiple predefined roles belonging to an application. An aggregated role can contain other aggregated roles. For example, a Shared Services Administrator or Provisioning Manager can create an aggregated role that combines the Planner and View User roles of a Planning application. Aggregating roles can simplify the administration of applications that have several granular roles. Global Shared Services roles can be included in aggregated roles. You cannot create an aggregated role that spans applications or products.

Users

User directories store information about the users who can access EPM System products. Both the authentication and the authorization processes use user information. You can create and manage Native Directory users only from Shared Services Console.

Users from all configured user directories are visible from Shared Services Console. These users can be individually provisioned to grant access rights on the EPM System applications registered with Shared Services. Oracle does not recommend provisioning individual users.

Groups

Groups are containers for users or other groups. You can create and manage Native Directory groups from Shared Services Console. Groups from all configured user directories are displayed in Shared Services Console. You can provision these groups to grant permissions for EPM System products registered with Shared Services.
Assumptions

- You have determined the deployment topology and identified the communication links that are to be secured using SSL.
- You have obtained the required certificates from a Certificate Authority (CA), either a well-known CA or your own, or created self-signed certificates. See “Required Certificates” on page 21.
- You are familiar with SSL concepts and procedures such as importing certificates. See “Information Sources” on page 19 for a list of reference documents.

Information Sources

SSL-enabling EPM System requires that you prepare components such as the application server, web server, databases, and user directories to communicate using SSL. This document assumes that you are familiar with the tasks involved in SSL-enabling these components.

- **Oracle WebLogic Server**: See “Configuring SSL” in the *Securing WebLogic Server Guide*.
- **Oracle HTTP Server**: See the following topics in the *Oracle HTTP Server Administrator’s Guide*:
  - Managing Security
Enabling SSL for Oracle HTTP Server

User Directories: See the documentation from the user directory vendor. Useful links:
- Oracle Internet Directory: See Oracle Internet Directory Administrator’s Guide
- Active Directory: See these documents:
  - Microsoft Windows Server 2008 Active Directory documentation
  - Microsoft Windows Server 2003 Active Directory documentation
- Novell eDirectory: Novell eDirectory documentation

Databases: See the documentation from the database vendor.

Internet Information Services: See How to Implement SSL in IIS.

Location References

This document refers to the following installation and deployment locations:

- **MIDDLEWARE_HOME** refers to the location of middleware components such as WebLogic Server, and, optionally, one or more **EPM_ORACLE_HOME**. The **MIDDLEWARE_HOME** is defined during EPM System product installation. The default **MIDDLEWARE_HOME** directory is Oracle/Middleware.

- **EPM_ORACLE_HOME** refers to the installation directory containing the files required to support EPM System products. The default **EPM_ORACLE_HOME** is **MIDDLEWARE_HOME**/EPMSystem11R1; for example, Oracle/Middleware/EPMSystem11R1.

  EPM System products are installed in the **EPM_ORACLE_HOME**/products directory; for example, Oracle/Middleware/EPMSystem11R1/products.

  Additionally, during EPM System product configuration, some products deploy components to **MIDDLEWARE_HOME**/user_projects/epmsystem1; for example, Oracle/Middleware/user_projects/epmsystem1.

- **EPM_ORACLE_INSTANCE** denotes a location that is defined during the configuration process where some products deploy components. The default location of **EPM_ORACLE_INSTANCE** is **MIDDLEWARE_HOME**/user_projects/epmsystem1; for example, Oracle/Middleware/user_projects/epmsystem1.

About SSL-Enabling EPM System Products

The EPM System deployment process automatically deploys Oracle's EPM System products to work in both SSL and non-SSL modes.

While specifying the common setting for EPM System, you specify whether to SSL-enable all server-to-server communication in your deployment.
Selecting SSL settings during the deployment process does not automatically configure your environment for SSL. It only sets a flag in the Oracle Hyperion Shared Services Registry to indicate that all EPM System components that use the Shared Services Registry must use the secure protocol (HTTPS) for server-to-server communication. You must complete additional procedures to SSL-enable your environment. These procedures are discussed in this document.

**Supported SSL Scenarios**

The following SSL scenarios are supported:

- SSL termination at the SSL offloader. See “Terminating SSL at the SSL Offloader” on page 22.
- Full SSL deployment. See “Full SSL Deployment of EPM System” on page 24.
- SSL termination at the web server. See “Terminating SSL at the Web Server” on page 35.

**Note:** This document assumes that you are using WebLogic Server to host web applications. If you are using WebSphere, refer to WebSphere documentation for information on SSL-enabling WebSphere application server and IBM HTTP Server plug-in.

**Required Certificates**

SSL communication uses certificates to establish trust between components. Oracle recommends that you use certificates from well-known third-party CAs to SSL-enable EPM System in a production environment.

**Note:** EPM System supports the use of wildcard certificates, which can secure multiple subdomains with one SSL certificate. Using a wildcard certificate can reduce management time and cost.

If you are using wildcard certificates to encrypt communication, you must disable host-name verification in WebLogic Server.

You require the following certificates for each server that hosts EPM System components:

- A root CA certificate.
Note: You need not install a root CA certificate in the Java keystore if you are using certificates from a well-known third-party CA whose root certificate is already installed in the Java keystore.

Firefox and Internet Explorer are preloaded with certificates of well-known third-party CAs. If you are acting as your own CA, you must import your CA root certificate into the keystore used by the clients accessed from such browsers. For example; if you are acting as your own CA, Oracle Hyperion Web Analysis clients cannot establish an SSL handshake with the server if your CA root certificate is not available to the browser from which Web Analysis is accessed.

- Signed certificates for each Oracle HTTP Server in your deployment.
- A signed certificate for WebLogic Server host machine. Managed servers on this machine can also use this certificate.
- Two certificates for the SSL offloader/load balancer. One of these certificates is for external communication and the other is for internal communication.

**Terminating SSL at the SSL Offloader**

**Subtopics**

- Deployment Architecture
- Assumptions
- Configuring EPM System
- Testing the Deployment

**Deployment Architecture**

In this scenario, SSL is used to secure the communication link between EPM System clients (for example, a browser) and SSL Offloader. The illustrated concept:
Assumptions

Subtopics

- SSL Offloader and Load Balancer
- Virtual Hosts

SSL Offloader and Load Balancer

A fully configured SSL offloader with a load balancer must be present in the deployment environment.

The load balancer must be configured to forward all requests received by the virtual hosts to Oracle HTTP Servers.

Virtual Hosts

SSL terminated at SSL offloader configuration uses two server aliases; for example, epm.myCompany.com and empinternal.myCompany.com, on the SSL offloader/load balancer, one for external communication between the offloader and browsers, and the other for internal communication among EPM System servers. Ensure that the server aliases point to the IP address of the machine, and that they are resolvable through DNS.

A signed certificate to support external communication between the offloader and browsers (through epm.myCompany.com) must be installed on the offloader/load balancer.
Configuring EPM System

The default deployment of EPM System components supports SSL termination at the SSL offloader. No additional action is required.

While configuring EPM System, ensure that the logical web applications point to the alias (for example, empinternal.myCompany.com) that was created for internal communication. See the following information sources to install and configure EPM System:

- Oracle Enterprise Performance Management System Installation and Configuration Guide
- Oracle Enterprise Performance Management System Installation Start Here
- Oracle Enterprise Performance Management System Installation and Configuration Troubleshooting Guide

Testing the Deployment

After completing the deployment process, verify that everything works by connecting to the secure Oracle Hyperion Enterprise Performance Management Workspace URL:

https://virtual_host_external:SSL_PORT/workspace/index.jsp

For example, https://epm.myCompany.com:443/workspace/index.jsp where 443 is the SSL port.

Full SSL Deployment of EPM System

Subtopics

- Deployment Architecture
- Assumptions
- Configuring EPM System for Full SSL

Deployment Architecture

In full SSL mode, communication across all securable channels is secured using SSL. This EPM System deployment scenario is the most secure.

The illustrated concept:
Note: Not all EPM System components can be SSL-enabled. Typically, back-end servers – for example, Oracle Hyperion Strategic Finance Server, and Financial Management Server – cannot be SSL-enabled.

Assumptions

Subtopics

- Databases
- EPM System
- SSL Offloader and Load Balancer

Databases

The database servers and clients are SSL-enabled. See your database documentation for information on SSL-enabling the database server and client.
EPM System

EPM System components, including WebLogic Server and Oracle HTTP Server, are installed and deployed. Further, your EPM System environment has been tested to ensure that everything is working in non-SSL mode. See the following information sources:

- Oracle Enterprise Performance Management System Installation and Configuration Guide
- Oracle Enterprise Performance Management System Installation Start Here
- Oracle Enterprise Performance Management System Installation and Configuration Troubleshooting Guide

If you plan to SSL-enable the database connections, during the configuration process, you must select the **Advanced Options** link in each database configuration screen, and then specify the required settings, which include the following:

- Select **Use secure connection to the database (SSL)** and enter a secure database URL; for example, `jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=myDBhost)(PORT=1529))(CONNECT_DATA=(SERVICE_NAME=myDBhost.myCompany.com)))`
- **Trusted Keystore**
- **Trusted Keystore Password**

See the Oracle Enterprise Performance Management System Installation and Configuration Guide for details.

SSL Offloader and Load Balancer

A fully configured SSL offloader with a load balancer must be present in the deployment environment.
Full SSL configuration uses two server aliases, for example, epm.myCompany.com and empinternal.myCompany.com, on the SSL offloader. One is for external communication between the offloader and browsers, and the other is for internal communication among EPM System servers. Ensure that the server aliases point to the IP address of the machine, and that they are resolvable through DNS.

The load balancer must be configured to forward all requests received by the virtual hosts to Oracle HTTP Servers.

The two signed certificates—one to support external communication between the offloader and browsers (through epm.myCompany.com), and the other to support internal communication (through empinternal.myCompany.com) among applications—must be installed on the offloader/load balancer. Oracle recommends that these certificates be tied to server aliases to prevent the exposure of server names and to enhance security.

**Configuring EPM System for Full SSL**

**Subtopics**

- Reconfiguring EPM System Common Settings
- Optional: Installing Root CA Certificate for WebLogic Server
- Installing Certificate on the WebLogic Server
- Configuring WebLogic Server
- Oracle HTTP Server Procedures
- Restarting Servers and EPM System
- Testing the Deployment
- Configuring SSL-enabled External User Directories

**Reconfiguring EPM System Common Settings**

During this process, you select the settings that force EPM System components to use SSL communication.

To reconfigure EPM System for SSL:

2. Clear Uncheck All.
3. Expand Hyperion Foundation configuration task, and then select Configure Common Settings.
4. Click Next until Configure Common Settings is displayed.
5. In the Configure Common Settings screen, select settings:

   **Caution!** Before selecting the settings to use SSL to communicate with the email server, ensure that the email server is configured for SSL.
a. Select **Use SSL for Web application server communication (Requires manual configuration)** to specify that EPM System should use SSL for communication.

b. Optional: Enter information in **Mail Server Host** and **Port**. To support SSL communication, you must specify the secure port used by the SMTP mail server.

c. Optional: To support SSL communication with the SMTP mail server, select **Use SSL to communicate with mail server**.

d. Select or enter settings in the remaining fields.

6. Click **Next** in subsequent EPM System Configurator screens after entering the required data.

7. When the deployment process is complete, the Summary screen is displayed. Verify that all the tasks completed successfully, and then click **Finish**.

**Optional: Installing Root CA Certificate for WebLogic Server**

The root certificates of most well-known third-party CAs are already installed in the Sun and JRockit JVM keystores. Complete the procedures in this section if you are not using certificates from a well-known third-party CA (not recommended). Default JVM keystore locations:

- **Sun JVM keystore**: `MIDDLEWARE_HOME/jdk160_29/jre/lib/security/cacerts`
- **JRockit JVM keystore**: `MIDDLEWARE_HOME/jrockit_160_29/jre/lib/security/cacerts`

**Note**: Perform this procedure on each EPM System server.
To install the root CA certificate:

1. Copy the root CA certificate into a local directory on the machine where WebLogic Server is installed.

2. From a console, change directory to $MIDDLEWARE_HOME/jdk160_29/jre/bin.

3. Execute a keytool command such as the following to install the signed certificate into the Sun JVM keystore:

   ```
   keytool -import -alias ALIAS -file CA_CERT_FILE -keystore KEYSTORE -storepass KEYSTORE_PASSWORD -trustcacerts
   ```

   For example, you can use the following command to add a certificate $CAcert.crt$ stored in the current directory into the Sun JVM keystore with $Blister$ as the certificate alias in the keystore. Default storepass (changeit) is assumed.

   ```
   keytool -import -alias Blister -file $CAcert.crt$ -keystore ./lib/security/cacerts -storepass changeit -trustcacerts
   ```

   **Note:** The preceding command and example use some of the syntax for importing certificates using keytool. See keytool documentation for a complete list of import syntax.

4. Execute a command such as the following to install the root CA certificate into the JRockit JVM keystore:

   ```
   keytool -import -alias ALIAS -file CERT_FILE -keystore KEYSTORE -storepass KEYSTORE_PASSWORD -trustcacerts
   ```

   For example, you can use the following command to add a certificate $CAcert.crt$ stored in the current directory into the JRockit JVM keystore with $Blister$ as the certificate alias. Default storepass (changeit) is assumed.

   ```
   keytool -import -alias Blister -file $CAcert.crt$ -keystore $MIDDLEWARE_HOME/jrockit_160_05/jre/lib/security/cacerts -storepass changeit -trustcacerts
   ```

   **Note:** Ensure that you replace $MIDDLEWARE_HOME$ with the directory path.

### Installing Certificate on the WebLogic Server

The default WebLogic Server installation uses a demo certificate to support SSL. Oracle recommends that you install a certificate from a well-known third-party to strengthen the security of your environment.

On each machine that hosts WebLogic Server, use a tool (for example, keytool) to create a custom keystore to store the signed certificate for WebLogic Server and EPM System web components.

To create a custom keystore and import certificate:

1. From a console, change directory to $MIDDLEWARE_HOME/jdk160_29/jre/bin$.

2. Execute a keytool command such as the following to create the custom keystore (identified by the -keystore directive in the command) in an existing directory:

   ```
   keytool -genkey -dname "cn=myserver, ou=EPM, o=myCompany, c=US" -alias epm_ssl -keypass password -keystore C:\oracle\Middleware\EPMSYstem11R1\ssl\keystore -storepass password -validity 365 -keyalg RSA
   ```
Note: The common name (cn) that you set must match the server name. If you use fully qualified domain name (FQDN) as the cn, you must use the FQDN while deploying web components.

3 Generate a certificate request.
keytool -certreq -alias epm_ssl -file C:/certs/epmssl_csr -keypass password -storetype jks -keystore C:\oracle\Middleware\EPMSystem11R1\ssl\keystore -storepass password

4 Obtain a signed certificate for the WebLogic Server machine.

5 Import the signed certificate into the keystore:
keytool -import -alias epm_ssl -file C:/certs/epmssl_crt -keypass password -keystore C:\Oracle\Middleware\EPMSystem11R1\ssl\keystore -storepass password

Configuring WebLogic Server

After deploying EPM System web components, you must configure them for SSL communication.

To configure the web components for SSL:
1 Start the WebLogic Server by executing a file stored in MIDDLEWARE_HOME/user_projects/domains/EPMSystem/bin:
   - startWebLogic.cmd (Windows)
   - startWebLogic.sh (UNIX)

2 Launch the WebLogic Server Administration Console by accessing the following URL:
   http://SERVER_NAME:Port/console

   For example, to access the WebLogic Server console deployed to the default port on myServer, you should use http://myServer:7001/console.

3 On the Welcome screen, enter the WebLogic Server administrator user name and password that you specified in EPM System Configurator.

4 In Change Center, click Lock & Edit.

5 In the left pane of the console, expand Environment, and then select Servers.

6 In the Summary of Servers screen, click the name of the server that you want to SSL-enable.

   For example, to SSL-enable Foundation Services components, you work with the EPMServer0 server.

7 Clear Listen Port Enabled to disable the HTTP listen port.

8 Ensure that SSL Listen Port Enabled is selected.

9 In SSL Listen Port, enter the SSL listen port where this server should listen for requests.

10 To specify the identity and trust keystores to use, select Keystores to open the Keystores tab.

11 Click Change.

12 Select an option:
- **Custom Identity and Custom Trust** if you are not using a server certificate from a well-known third-party CA
- **Custom Identity and Java Standard Trust** if you are using a server certificate from a well-known third-party CA

13 Click **Save**.

14 In **Custom Identity Keystore**, enter the path of the keystore where the signed WebLogic Server certificate is installed.

15 In **Custom Identity Keystore Type**, enter **jks**.

16 In **Custom Identity Keystore Passphrase** and **Confirm Custom Identity Keystore Passphrase**, enter the keystore password.

17 If you selected **Custom Identity and Custom Trust** in Keystores:
   a. In **Custom Trust Keystore**, enter the path of the custom keystore where the root certificate of the CA that signed your server certificate is available.
   b. In **Custom Trust Keystore Type**, enter **jks**.
   c. In **Custom Trust Keystore Passphrase** and **Confirm Custom Trust Keystore Passphrase**, enter the keystore password.

18 Click **Save**.

19 Specify SSL settings.
   a. Select **SSL**.
   b. In **Private Key Alias**, enter the alias that you specified while importing the signed WebLogic Server certificate.
   c. In **Private Key Passphrase** and **Confirm Private Key Passphrase**, enter the password to be used to retrieve the private key.
   d. Click **Save**.

20 Complete step 6 through step 19 for each managed server belonging to this host.

21 In **Change Center**, click **Activate Changes**.

---

**Oracle HTTP Server Procedures**

**Subtopics**

- Creating a Wallet and Installing Certificate for Oracle HTTP Server
- SSL-Enabling Oracle HTTP Server

**Creating a Wallet and Installing Certificate for Oracle HTTP Server**

A default wallet is automatically installed with Oracle HTTP Server. You must configure a real wallet for each Oracle HTTP Server in your deployment.
To create and install Oracle HTTP Server certificate:

1. On each machine that hosts Oracle HTTP Server, launch the Wallet Manager.
   - Windows: Select Start, then All Programs, Oracle-OHxxxxx, then Integrated Management Tools, and then Wallet Manager.
     
     xxxx is the Oracle HTTP Server instance number.
   - UNIX: Execute MIDDLEWARE_HOME/ohs/bin/owm to launch the Wallet Manager from the command line.

   **Note:** The Wallet Manager requires a graphical environment.

2. Create a new, empty Wallet.
   
   a. In Oracle Wallet Manager, select Wallet, and then New.
   
   b. Click Yes to create a default wallet directory, or No to create the Wallet file in a location of your choice.
   
   c. In Wallet Password and Confirm Password on the New Wallet screen, enter the password that you want to use.
   
   d. Click OK.
   
   e. In the confirmation dialog box, click No.

3. Optional: If you are not using a CA that is known to Oracle HTTP Server, import the root CA certificate into the Wallet.
   
   a. In Oracle Wallet Manager, right-click Trusted Certificates and select Import Trusted Certificate.
   
   b. Browse and select the root CA certificate.
   
   c. Select Open.

4. Create a certificate request.
   
   
   b. In Create Certificate Request, enter the required information.
      
      For the common name, enter the fully qualified server alias; for example, epm.myCompany.com or epminternal.myCompany.com, available in the hosts file on your system.
   
   c. Click OK.
   
   d. In the confirmation dialog box, click OK.
   
   e. Right-click the certificate request that you created, and then select Export Certificate Request.

   f. Specify a name for the certificate request file.

5. Using the certificate request files, obtain signed certificates from the CA.

6. Import signed certificates.
a. In Oracle Wallet Manager, right-click the certificate request that was used to obtain the signed certificate, and then select **Import User Certificate**.

b. In Import Certificate, click **OK** to import the certificate from a file.

c. In Import Certificate, select the Certificate file, and then click **Open**.

7 **Save the Wallet to a convenient location; for example, EPM_ORACLE_INSTANCE/httpConfig/ohs/config/OHS/ohs_component/keystores/epmsystem.**

8 **Select Wallet, and then Auto Login to activate auto login.**

### SSL-Enabling Oracle HTTP Server

After reconfiguring the web server on each machine that hosts Oracle HTTP Server, update Oracle HTTP Server configuration file by replacing the location of the default Wallet with the location of the wallet that you created.

To configure Oracle HTTP Server for SSL:

1 **Reconfigure the web server on each Oracle HTTP Server host machine in your deployment.**
   a. Start EPM System Configurator for the instance.
   b. In configuration task selection screen, complete these steps, and then click **Next**.
      i. Clear the selection (uncheck) from **Uncheck All**.
      ii. Expand **Hyperion Foundation** task group, and select (check) **Configure Web Server**.
   c. In Configure Web Server, click **Next**.
   d. In Confirmation, click **Next**.
   e. In Summary, click **Finish**.

2 **Update the configuration settings of each Oracle HTTP Server in your deployment.**
   a. Using a text editor, open `EPM_ORACLE_INSTANCE/httpConfig/ohs/config/OHS/ohs_component/ssl.conf`.
   b. Locate the `SSLWallet` directive and change its value so that it points to the wallet where you installed the certificate. If you created the wallet in `EPM_ORACLE_INSTANCE/httpConfig/ohs/config/OHS/ohs_component/keystores/epmsystem`, your `SSLWallet` directive may be as follows:

   `SSLWallet "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/${COMPONENT_NAME}/keystores/epmsystem"`

c. Save and close `ssl.conf`.

3 **Update mod_wl_ohs.conf on each Oracle HTTP Server in your deployment.**
   a. Using a text editor, open `EPM_ORACLE_INSTANCE/httpConfig/ohs/config/OHS/ohs_component/mod_wl_ohs.conf`.
   b. Ensure that the `WLSSLWallet` directive points to the Oracle Wallet where the SSL certificate is stored.

   `WLSSLWallet MIDDLEWARE_HOME/ohs/bin/wallets/myWallet`
For example, C:/Oracle/Middleware/ohs/bin/wallets/myWallet

c. Save and close mod_wl_ohs.conf.

Restarting Servers and EPM System

Restart all the servers in the deployment, and then start EPM System on each server.

Testing the Deployment

After completing the SSL deployment, verify that everything works.

➢ To test your deployment:

1. Using a browser, access the secure EPM Workspace URL:

   If you used epm.myCompany.com as the server alias for external communication and 4443 as the SSL port, the EPM Workspace URL is

   https://epm.myCompany.com:4443/workspace/index.jsp

2. On the Logon screen, enter a user name and password.

3. Click Log On.

4. Verify that you can securely access the deployed EPM System components.

Configuring SSL-enabled External User Directories

Subtopics

- Assumptions
- Import the root CA certificate
- Configure External User Directories

Assumptions

- The external user directories that you plan to configure in Shared Services Console are SSL-enabled.
- If you did not use a certificate from a well-known third-party CA to SSL-enable the user directory, you have a copy of the root certificate of the CA that signed the server certificate.

Import the root CA certificate

If you did not use a certificate from a well-known third-party CA to SSL-enable the user directory, you must import the root certificate of the CA that signed the server certificate into the following JVMs:

Use a tool such, as keytool, to import the root CA certificate.

- All EPM System servers:
The keystore used by the JVM on each EPM System component host machine. By default, EPM System components use the following keystore:

`MIDDLEWARE_HOME/jdk160_11/jre/lib/security/cacerts`

**Configure External User Directories**

You configure user directories using the Shared Services Console. While configuring user directories, you must select the **SSL Enabled** option that instructs EPM System security to use the secure protocol to communicate with the user directory. You can SSL-enable a connection between EPM System security and LDAP-enabled user directories; for example, Oracle Internet Directory and Microsoft Active Directory.


**Terminating SSL at the Web Server**

**Subtopics**

- Deployment Architecture
- Assumptions
- Configuring EPM System
- Testing the Deployment

**Deployment Architecture**

In this scenario, SSL is used to secure the communication link between EPM System clients (for example, a browser) and Oracle HTTP Server. The illustrated concept:

- **Sun JVM keystore**: `MIDDLEWARE_HOME/jdk160_11/jre/lib/security/cacerts`
- **JRockit JVM keystore**: `MIDDLEWARE_HOME/jrockit_160_05/jre/lib/security/cacerts`
Assumptions

This configuration uses two server aliases; for example, epm.myCompany.com and empinternal.myCompany.com, on the web server, one for external communication between the web server and browsers, and the other for internal communication among EPM System servers. Ensure that the server aliases point to the IP address of the machine, and that they are resolvable through DNS.

A signed certificate to support external communication with browsers (for example, through epm.myCompany.com) must be installed on the web server (where the virtual host that supports secure external communication is defined). This virtual host should terminate SSL and forward HTTP requests to the Oracle HTTP Server.

Configuring EPM System

The default deployment of EPM System components supports SSL termination at the web server. No additional action is required.

While configuring EPM System, ensure that the logical web applications point to the virtual host (for example, empinternal.myCompany.com) that was created for internal communication. For detailed information, see “Oracle Enterprise Management System Installation and Configuration Guide.”
Testing the Deployment

After completing the deployment process, verify that everything works by connecting to the secure EPM Workspace URL:

https://virtual_host_external:SSL_PORT/workspace/index.jsp

For example, https://epm.myCompany.com:443/workspace/index.jsp where 443 is the SSL port.

Enabling Encryption for Financial Reporting Studio

To configure Oracle Hyperion Financial Reporting Studio for encrypted RMI communication, add the following to the JVM startup parameters (shell script files in UNIX servers) or JVMOption Windows registry entries (Windows servers).

-Djavax.net.ssl.trustStore=TRUSTSTORE_LOCATION

Replace TRUSTSTORE_LOCATION with the absolute location of the keystore where you installed the CA root certificate.

The registry location for adding this parameter for Financial Reporting Studio on a Windows server is HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\Hyperion Reports \HReports\JVM.

The location for adding JVM parameters for the Financial Reporting web application is HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions \FinancialReporting0\HyS9FRReports.

SSL for Essbase

Subtopics

- Overview
- Default Deployment
- Required Certificates and Their Location
- Essbase and SSL-Enabled EPM System
- Installing and Deploying Essbase Components
- Using Trusted Third-Party CA Certificates for Essbase
- Establishing a Per-Session SSL Connection

Overview

Essbase supports one-way SSL only, in which the Essbase instance (server and agent) is secured using certificates.
This section explains the procedures for replacing the default certificates that are used to secure communication between an Essbase instance and components such as MaxL, Oracle Essbase Administration Services Server, Oracle Essbase Studio Server, Oracle Hyperion Provider Services, Foundation Services, Planning, Financial Management, and Shared Services Registry.

**Default Deployment**

Essbase can be deployed to work in SSL and non-SSL modes. Essbase Agent listens on a non-secure port; it also can be configured to listen on a secure port. All connections accessing the secure port are treated as SSL connections. If a client connects to the Essbase Agent on the non-SSL port, the connection is treated as a non-SSL connection. Components can establish concurrent non-SSL and SSL connections to an Essbase Agent.

You can control SSL on a per-session basis by specifying the secure protocol and port when you log in. See “Establishing a Per-Session SSL Connection” on page 45.

If SSL is enabled, all communication within an Essbase instance is encrypted to ensure data security.

Default deployments of Essbase components in secure mode uses self-signed certificates to enable SSL communication, mainly for testing purposes. Oracle recommends that you use certificates from well-known third-party CAs to SSL-enable Essbase in production environments.
Typically, an Oracle Wallet stores the certificate that enables SSL communication with clients that use Essbase RTC (C APIs) and a Java keystore stores the certificate that enables SSL communication with components that utilize JAPI for communication. To establish SSL communication, Essbase clients and tools store the root certificate of the CA that signed the Essbase Server and Agent certificates. See “Required Certificates and Their Location” on page 39.

**Required Certificates and Their Location**

Oracle recommends the use of certificates from well-known third-party CAs to SSL-enable Essbase in a production environment. You may use the default self-signed certificates for test purposes.

*Note:* Essbase supports the use of wildcard certificates, which can secure multiple subdomains with one SSL certificate. Using a wildcard certificate can reduce management time and cost.

Wildcard certificates cannot be used if host-name check is enabled.

You require the following certificates:

- A root CA certificate.

Components that use Essbase RTC (C APIs) to establish a connection to Essbase require that the root CA certificate be stored in Oracle Wallet. Components that use JAPI to establish a connection require that the root CA certificate be stored in a Java keystore. The required certificate and their location are indicated in the following table.

*Note:* You may not need to install root CA certificate if you are using certificates from a well-known third-party CA whose root certificate is already installed in Oracle Wallet.

- Signed certificate for Essbase Server and Essbase Agent.

<table>
<thead>
<tr>
<th>Component</th>
<th>Keystore</th>
<th>Certificate</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxL</td>
<td>Oracle Wallet</td>
<td>root CA certificate</td>
</tr>
<tr>
<td>Administration Services Server</td>
<td>Oracle Wallet</td>
<td>root CA certificate</td>
</tr>
<tr>
<td>Provider Services</td>
<td>Oracle Wallet</td>
<td>root CA certificate</td>
</tr>
<tr>
<td>EPM System Database</td>
<td>Oracle Wallet</td>
<td>root CA certificate</td>
</tr>
<tr>
<td>Essbase Studio Server</td>
<td>Java Keystore</td>
<td>root CA certificate</td>
</tr>
<tr>
<td>Planning</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Oracle Wallet</td>
<td>root CA certificate</td>
</tr>
<tr>
<td></td>
<td>• Java Keystore</td>
<td>root CA certificate</td>
</tr>
<tr>
<td>Financial Management</td>
<td>Java Keystore</td>
<td>root CA certificate</td>
</tr>
</tbody>
</table>
### Component | Keystore | Certificate
--- | --- | ---
Essbase (Server and Agent) | Oracle Wallet, Java Keystore | root CA certificate, Signed certificate for Essbase Server and Agent
Shared Services Repository |  | 

1 You require only one instance of the keystore to support multiple components that use similar keystore.

2 Multiple components can use a root certificate installed in a keystore.

3 Certificates must be installed in the default Oracle Wallet and the Java keystore.

---

## Essbase and SSL-Enabled EPM System

Securing EPM System using SSL does not SSL-enable Essbase.

The only setting that affects an Essbase instance that is deployed with SSL-enabled EPM System is the JDBC connection setting stored in the Shared Services Registry. If EPM System web components are configured to use a secure JDBC connection to communicate with the Foundation Services database, the Shared Services Registry contains a secure JDBC connection string. In this scenario, manually install the root CA certificate used by Essbase on the database server.

See your database documentation for information on SSL-enabling the database server and client.

## Installing and Deploying Essbase Components

The configuration process allows you to select a secure agent port (default is 6423), which you can change when configuring Essbase. By default, the deployment process installs the required self-signed certificates to create a functional secure deployment for testing.

The Oracle Hyperion Enterprise Performance Management System Installer installs an Oracle Wallet and self-signed certificate within `ARBOR_PATH` on the machine that hosts the Essbase instance if Oracle HTTP Server is installed. In single host deployments, all Essbase components share this certificate.
Using Trusted Third-Party CA Certificates for Essbase

Subtopics

- Creating Certificate Requests and Obtaining Certificates
- Obtaining and Installing Root CA certificate
- Installing Signed Certificates
- Updating Default Settings

Creating Certificate Requests and Obtaining Certificates

Generate a certificate request to obtain a certificate for the server that hosts Essbase Server and Essbase Agent. A certificate request contains encrypted information specific to your Distinguished Name (DN). You submit the certificate request to a signing authority to obtain an SSL certificate.

You use a tool such as keytool or Oracle Wallet Manager to create a certificate request. For detailed information on creating a certificate request, see the documentation of the tool that you are using.

If you are using keytool, use a command such as the following to create a certificate request:

```
keytool -certreq -alias essbase_ssl -file C:/certs/essabse_server_csr -keypass password -storetype jks -keystore C:\oracle\Middleware\EPMSys\11.11.1\Essbase_ssl\keystore -storepass password
```

Obtaining and Installing Root CA certificate

The root CA certificate attests to the validity of the certificate that is used to support SSL. It contains the public key against which the private key that was used to sign the certificate is matched to verify the certificate. You can obtain the root CA certificate from the certificate authority that signed your SSL certificates.

Install the root certificate of the CA that signed the Essbase Server certificate on clients that connect to the Essbase Server or Agent. Ensure that the root certificate is installed in the keystore appropriate for the client. See “Required Certificates and Their Location” on page 39.

Note: Multiple components can use a root CA certificate installed on a server machine.

Oracle Wallet

Refer to Table 1, “Required Certificates and Their Locations” for a list of components that require the CA root certificate in an Oracle Wallet. You can create a wallet or install the certificate in the demo wallet where the default self-signed certificate is installed.

See Oracle Wallet Manager documentation for detailed procedures to create wallets and to import root CA certificate.
Java Keystore

Refer to Table 1, “Required Certificates and Their Locations” for a list of components that require the root CA certificate in an Java keystore. You can add the certificate into the keystore where the default self-signed certificate is installed or create a keystore for storing the certificate.

**Note:** The root CA certificates of many well-known third-party CAs are already installed in the Java keystore.

Refer to the documentation of the tool you are using for detailed instructions. If you are using keytool, use a command, such as the following, to import the root certificate:

```
keytool -import -alias blister_CA -file c:/certs/CA.crt -keypass password -trustcacerts -keystore C:\Oracle\Middleware\EPMSystem11R1\Essbase_ssl\keystore -storepass password
```

Installing Signed Certificates

You install the signed SSL certificates on the server that hosts Essbase Server and Essbase Agent. Components that use Essbase RTC (C APIs) to establish a connection to Essbase Server or Agent require that the certificate be stored in an Oracle Wallet along with the root CA certificate. Components that use JAPI to establish a connection to Essbase Server or Agent require that the root CA certificate and signed SSL certificate be stored in a Java keystore. For detailed procedures, see these information sources:

- Oracle Wallet Manager documentation
- Documentation or online help of the tool; for example, keytool, that you use to import the certificate

If you are using keytool, use a command, such as the following, to import the certificate:

```
keytool -import -alias essbase_ssl -file C:/certs/essbase_ssl_crt -keypass password -keystore C:\Oracle\Middleware\EPMSystem11R1\Essbase_ssl\keystore -storepass password
```

Updating Default Settings

**Subtopics**

- **Updating Essbase SSL Settings**
- **Customizing SSL Properties of JAPI Clients**
- **Available Cipher Suites for Components that Use Essbase C APIs**

You customize the SSL settings for components that use C APIs (Essbase Server and clients) by specifying their value in `essbase.cfg`.

You customize Essbase Server SSL settings by specifying their value in `essbase.cfg`.
Updating Essbase SSL Settings

Edit essbase.cfg to customize Essbase SSL settings, such as:

- Setting to enable secure mode
- Setting to enable clear mode
- Preferred mode to communicate with clients (used by clients only)
- Secure port
- Cipher suites
- Oracle Wallet path

To update essbase.cfg:

1. **Using a text editor**, open `EPM_ORACLE_INSTANCE/EssbaseServer/essbaseserver1/bin/essbase.cfg`.
2. **Enter settings as needed. See Table 2.**

### Table 2: Essbase SSL Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnableClearMode²</td>
<td>Enables unencrypted communication between Essbase applications and Essbase Agent. If this property is set to FALSE, Essbase does not handle non-SSL requests.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>TRUE</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>EnableClearMode FALSE</td>
</tr>
<tr>
<td>EnableSecureMode</td>
<td>Enables SSL encrypted communication between Essbase clients and Essbase Agent. This property must be set to TRUE to support SSL.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>FALSE</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>EnableSecureMode TRUE</td>
</tr>
<tr>
<td>SSLCipherSuites</td>
<td>A list of cipher suites, in order of preference, to use for SSL communication. See “Available Cipher Suites for Components that Use Essbase C APIs” on page 45. Essbase Agent uses one of these cipher suites for SSL communication. The first cipher suite in the list is accorded the highest priority when the agent chooses a cipher suit.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>SSL_RSA_WITH_RC4_128_MD5</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>SSLCipherSuites SSL_RSA_WITH_AES_256_CBC_SHA, SSL_RSA_WITH_DES_CBC_SHA</td>
</tr>
<tr>
<td>AgentSecurePort</td>
<td>The secure port at which the agent listens.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>6423</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>AgentSecurePort 16001</td>
</tr>
<tr>
<td>WalletPath</td>
<td>Location of the Oracle Wallet (fewer than 1024 characters) that stores the root CA certificate and signed certificate.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>ARBORPATH/bin/wallet</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>WalletPath/usr/local/wallet</td>
</tr>
</tbody>
</table>
### Customizing SSL Properties of JAPI Clients

A number of default properties are set for you when you deploy Essbase components that rely on JAPI. These customizable properties are externalized in `essbase.properties`.

To update SSL properties of JAPI clients:

1. Using a text editor, open `EPM_ORACLE_HOME\common\EssbaseJavaAPI\11.1.2.0\bin\essbase.properties`.
2. Updates the properties as needed. See Table 3 for description of customizable JAPI client properties.

### Table 3  Default SSL properties for JAPI Clients

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>olap.server.ssl.alwaysSecure</td>
<td>Sets the mode that clients should use against all Essbase instances. Change this property value to true to enforce SSL mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> false</td>
</tr>
<tr>
<td>olap.server.ssl.securityHandler</td>
<td>Package name for handling the protocol. You can change this value to indicate another handler.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> java.protocol.handler.pkgs</td>
</tr>
<tr>
<td>olap.server.ssl.securityProvider</td>
<td>Oracle uses the Sun SSL protocol implementation. You can change this value to indicate another provider.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> com.sun.net.ssl.internal.www.protocol</td>
</tr>
<tr>
<td>olap.server.ssl.supportedCiphers</td>
<td>A comma-separated list of additional ciphers to be enabled for secure communication. You must specify only ciphers that Essbase supports. See &quot;Available Cipher Suites for Components that Use Essbase C APIs&quot; on page 45.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> SSL_RSA_WITH_AES_256_CBC_SHA, SSL_RSA_WITH_AES_128_CBC_SHA</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>olap.server.ssl.trustManagerClass</td>
<td>The TrustManager class to use to validate SSL certificate by verifying the signature and checking certificate expiration date. By default, this property is not set to enforce all verification checks. To not enforce verification checks, set the value of this parameter to <code>com.essbase.services.olap.security.EssDefaultTrustManager</code>, which is the default TrustManager class that allows all validation checks to succeed. To implement a custom TrustManager, specify a fully qualified class name of the TrustManager class that implements <code>javax.net.ssl.X509TrustManager</code> interface. <strong>Example:</strong> <code>com.essbase.services.olap.security.EssDefaultTrustManager</code></td>
</tr>
<tr>
<td>olap.server.ssl.keyManagerClass</td>
<td>This parameter is not used in this release.</td>
</tr>
</tbody>
</table>

3. **Save and close** `essbase.properties`.
4. **Restart all Essbase components**.

### Available Cipher Suites for Components that Use Essbase C APIs

These cipher suites are supported by the SSL implementation on Essbase Server:

- `SSL_RSA_WITH_AES_256_CBC_SHA`
- `SSL_RSA_WITH_AES_128_CBC_SHA`
- `SSL_RSA_WITH_3DES_EDE_CBC_SHA`
- `SSL_RSA_WITH_DES_CBC_SHA`
- `SSL_RSA_WITH_RC4_128_SHA`
- `SSL_RSA_WITH_RC4_128_MD5`

### Establishing a Per-Session SSL Connection

Essbase components; for example, MaxL, can control SSL at session level by connecting to Essbase Agent using `secure` as the transport keyword. For example, you can establish a secure connection between MaxL and Essbase Agent by executing one of the following commands from a MaxL Console:

```plaintext
login admin welcome1 on hostA:PORT:secure
login admin welcome1 on hostA:secure
```

Per-session control takes priority over configuration settings specified in `essbase.cfg`. If no transport keyword is specified, Essbase clients use the value set for `ClientPreferredMode` to determine whether to initiate a secure connection with Oracle Essbase. If `ClientPreferredMode` setting is not set to secure, the communication always occurs over a nonsecure channel.
Enabling SSO with Security Agents

In This Chapter

Supported SSO Methods .................................................................................. 47
Single Sign-on from Oracle Access Manager ............................................................ 49
OracleAS Single Sign-on .................................................................................. 51
Protecting EPM System Products for SSO ............................................................... 57
SiteMinder SSO ............................................................................................ 61
Kerberos Single Sign-on................................................................................... 64
Configuring the EPM System for SSO ................................................................. 74
Single Sign-on Options for Smart View ............................................................... 75

Supported SSO Methods

Subtopics

- HTTP Header
- Custom Login Class
- HTTP Authorization Header
- Get Remote User from HTTP Request

SSO requires that the web identity management solution pass the login name of authenticated users to EPM System products. You can use the following standard EPM System methods to integrate EPM System with commercial and custom Web-based SSO solutions.

- “HTTP Header” on page 48
- “Custom Login Class” on page 48
- “HTTP Authorization Header” on page 49
- “Get Remote User from HTTP Request” on page 49

Caution! As a security measure, Oracle recommends that you implement client certificate authentication (two-way SSL) between the web server and the application server if your organization uses methods that carry user identity in the header for identity propagation.
HTTP Header

If you are using Oracle Access Manager or SiteMinder (or a custom SSO provider) as the web identity management solution, use an HTTP header to pass the login name of the authenticated user to EPM System products.

The login name of an EPM System product user is determined by the Login Attribute that is specified while configuring user directories in Shared Services. See “Configuring OID, Active Directory, and Other LDAP-Based User Directories” in the Oracle Enterprise Performance Management System User and Role Security Guide for a brief description of the Login Attribute.

The HTTP header must contain the value of the attribute that is set as the Login Attribute. For example, if uid is the Login Attribute value, the HTTP header must carry the value of the uid attribute.

See your web identity management solution documentation for detailed information on defining and issuing custom HTTP headers.

EPM System security parses the HTTP header and validates the login name that it carries against the user directories configured on Shared Services.

Custom Login Class

When a user logs in, the web identity management solution authenticates the user against a directory server and encapsulates the credentials of the authenticated user in an SSO mechanism to enable SSO with downstream systems. If the web identity management solution uses a mechanism unsupported by EPM System products, or if the value of the Login Attribute is not available in the SSO mechanism, you can use a custom login class to derive and pass the value of the Login Attribute to EPM System products.

This method allows EPM System to integrate with security agents that use X509 certificate-based authentication. Using a custom login class as the authentication mechanism requires using standard Shared Services APIs to define the SSO interface between EPM System products and the web identity management solution. The custom login class must pass the value of the Login Attribute to EPM System products. See “Configuring OID, Active Directory, and Other LDAP-Based User Directories” in the Oracle Enterprise Performance Management System User and Role Security Guide for a brief description of Login Attribute. For sample code and implementation steps, see Appendix B, “Implementing a Custom Login Class”.

To use a custom login class, an implementation of com.hyperion.css.CSSSecurityAgentIF interface must be available in the classpath. CSSSecurityAgentIF defines the getter method for retrieving user name and password (optional). If the interface returns a null password, security authentication treats the provider as trusted and verifies the existence of the user in configured providers. If the interface returns a non-null value for password, EPM System attempts to authenticate the request using the user name and password returned by this implementation.

CSSSecurityAgentIF comprises two methods: getUserName and getPassword.
**getUserName Method**

This method returns the user name for authentication.

```java
java.lang.String getUserName(
    javax.servlet.http.HttpServletRequest req,
    javax.servlet.http.HttpServletResponse res)
throws java.lang.Exception
```

The `req` parameter identifies the HTTP request that carries the information that is used to determine the user name. The `res` parameter is not used (preset for backward compatibility).

**getPassword Method**

This method returns clear-text password for authentication. Password retrieval is optional.

```java
java.lang.String getPassword(
    javax.servlet.http.HttpServletRequest req,
    javax.servlet.http.HttpServletResponse res)
throws java.lang.Exception
```

The `req` parameter identifies the HTTP request that carries the information that is used to determine the password. The `res` parameter is not used (preset for backward compatibility).

**HTTP Authorization Header**

EPM System security supports the use of an HTTP authorization header to pass the value of the `Login Attribute` to EPM System products from web identity management solutions. EPM System products parse the authorization header to retrieve the user's login name.

**Get Remote User from HTTP Request**

EPM System security supports the use of an HTTP request to pass the value of `Login Attribute` to EPM System products from web identity management solutions. Use this SSO method if the web identity management solution passes an HTTP request containing the value of the `Login Attribute`, which is set using the `setRemoteUser` function.

This method is used for OracleAS Single Sign-on (OSSO) and Oracle Application Server integrated with Integrated Windows Authentication.

**Single Sign-on from Oracle Access Manager**

EPM System integrates with Oracle Access Manager by accepting a custom HTTP header (default name `HYPLOGIN`) that contains the login attribute value. The login attribute is set when you configure an external user directory in Shared Services. See “Configuring OID, Active Directory, and Other LDAP-Based User Directories” in the *Oracle Enterprise Performance Management System User and Role Security Guide* for a brief description of `Login Attribute`.
You can use any header name that provides the value of login attribute to EPM System. You use the header name while configuring Shared Services for SSO from Oracle Access Manager.

EPM System uses the value of the login attribute to authenticate the user against a configured user directory (in this case, the user directory against which Oracle Access Manager authenticates users) and then generates an EPM SSO token that enables SSO across EPM System. Provisioning information of the user is checked in Native Directory to authorize the user to EPM System resources.

**Note:** Administration Services console, which is a thick client, does not support SSO from Oracle Access Manager.

Information about configuring Oracle Access Manager and performing tasks such as setting up the HTTP header and policy domains is available in the Oracle Access Manager documentation. This guide assumes a working Oracle Access Manager deployment where you have completed the following tasks:

- Setup the required policy domains for EPM System components.
- Configured an HTTP header to pass login attribute value to EPM System.
- Protected the EPM System resources listed in “Resources to Protect” on page 57. Requests to access protected resources are challenged by Oracle Access Manager.
- Unprotected the EPM System resources listed in “Resources to Unprotect” on page 58. Requests to access unprotected resources are not challenged by Oracle Access Manager.

To configure EPM System for SSO from Oracle Access Manager:

1. **Add the user directory that Oracle Access Manager uses to authenticate users as an external user directory in EPM System.** See “Configuring OID, Active Directory, and Other LDAP-Based User Directories” in the Oracle Enterprise Performance Management System User and Role Security Guide.

   **Note:** Ensure that the **Trusted** check box in the Connection Information screen is selected to indicate that the user directory is a trusted SSO source.

2. **Configure EPM System for SSO.** See “Configuring the EPM System for SSO” on page 74.

   Select Oracle Access Manager from the **SSO Provider or Agent** list. If the HTTP header from Oracle Access Manager uses a name other than **HYPLOGIN**, enter the name of the custom header in the text box next to the **SSO Mechanism** list.

3. **Oracle Hyperion Data Relationship Management only:**

   a. Configure Data Relationship Management for Shared Services authentication.
   
   b. Enable SSO in Data Relationship Management Console.

      See Data Relationship Management documentation for detailed information.
OracleAS Single Sign-on

The OracleAS Single Sign-on (OSSO) solution provides SSO access to web applications using Oracle Internet Directory (OID) as the user directory. Users use a user name and password defined in an OID to log in to EPM System products.

Process Flow

The OSSO process:

1. Using an EPM System URL, for example, http://OSSO_OHS_Server_NAME:OSSO_OHS_Server_PORT/interop/index.jsp, users access an EPM System component that is defined as an OSSO protected application.

2. Because the URL is under OSSO protection, mod_osso, deployed on Oracle HTTP Server, intercepts the request. Using mod_osso, Oracle HTTP Server checks for a valid cookie. If a valid cookie is not available in the request, Oracle HTTP Server redirects users to the OSSO Server, which challenges users for credentials, which it authenticates against OID.

3. OSSO Server creates the obSSOCookie and returns control to the mod_osso module on the Oracle HTTP Server, which sets the obSSOCookie in the browser. It also redirects the request to the EPM System resource through mod_wl_ohs (WebLogic Server) or mod_proxy (IIS Server). Before forwarding the request to an EPM System resource, Oracle HTTP Server sets the Proxy-Remote-User header which EPM System security uses to enable SSO.
4. The EPM System component verifies that the user whose identity is retrieved from Proxy-Remote-User is present in OID. For this process to work, the OID that is configured with the OSSO Server should be configured as an external user directory in Shared Services.

**Prerequisites**

1. A fully functional Oracle Application Server Infrastructure.
   
   To establish an Oracle Application Server Infrastructure, install and configure Oracle Identity Management Infrastructure 10.1.4. Ensure that OSSO is enabled. Oracle Identity Management Infrastructure 10.1.4 installation includes the following components to support OSSO.
   
   - Oracle 10g OSSO Server.
   - An OID, which the OSSO Server uses to validate credentials. See the following guides:
     - Oracle Fusion Middleware Installation Guide for Oracle Identity Management
     - Oracle Fusion Middleware Administrator’s Guide for Oracle Internet Directory
   - Oracle HTTP Server as a front-end to the OSSO Server. This installation includes `mod_osso` that allows you to define partner applications for OSSO.

   **Note:** This Oracle HTTP Server instance is a part of the OSSO infrastructure; it is not directly used for configuring OSSO for EPM System components.

   During the installation process ensure that `mod_osso` is registered with the OSSO Server as a partner application.

   
   When you configure the web server for EPM System components, EPM System Configurator configures the following on the Oracle HTTP Server to proxy requests to the application server:
   
   - `mod_wl_ohs.conf` to proxy requests to WebLogic Server
   - `mod_proxy` to proxy requests to IIS Server
Enabling OSSO for EPM System

Subtopics

- Registering EPM System Web Server as a Partner Application
- Optional: Defining Virtual Host
- Creating mod_osso.conf
- Relocating osso.conf
- Adding Cache Management Configuration for Reporting and Analysis
- Configuring EPM System for OSSO
- Optional: Enabling Debug Messages on OSSO Server
- Optional: Enabling Debug Messages for Protected Resources

This section assumes that you have a fully configured OSSO infrastructure. See the Oracle Application Server Administrator’s Guide.

Registering EPM System Web Server as a Partner Application

You use the Oracle Identity Manager SSO registration tool (ssoreg.sh or ssoreg.bat) to register EPM System web server as a partner application on the Oracle HTTP Server that front-ends the OSSO Server.

Perform this procedure on the server that hosts the Oracle HTTP Server that front-ends the OSSO Server. This process generates and stores an obfuscated osso.conf in the location of your choice.

To register EPM System web server as a partner application:

1. Open a console on the server that hosts the Oracle HTTP Server that front-ends the OSSO Server and navigate to ORACLE_HOME/sso/bin directory of Oracle HTTP Server, for example to C:/OraHome_1/sso/bin (Windows).
2. Execute a command similar to the following with -remote_midtier option:

   ssoreg.bat -site_name epm.myCompany.com
   -mod_osso_url http://epm.myCompany.com:19400
   -config_mod_osso TRUE
   -update_mode CREATE
   -remote_midtier
   -config_file C:\OraHome_1\myFiles\osso.conf

The following explains the parameters used in this command. In this description, partner application refers to the Oracle HTTP Server that is used as the EPM System web server.

- **-site_name** identifies the web site of the partner application; for example, epm.myCompany.com.
- **-mod_osso_url** indicates the partner application URL, in PROTOCOL://HOST_NAME:PORT format. This is the URL at which the EPM System web server accepts incoming client requests, for example, http://epm.myCompany.com:19000.
- **-config_mod_osso** identifies that the partner application uses mod_osso. You must include the config_mod_osso parameter to generate osso.conf.
- **update_mode** indicates the update mode. Use CREATE, the default, to generate a new record.

- **remote_midtier** specifies that the mod_osso partner application is at a remote mid-tier. Use this option when the partner application is at a different ORACLE_HOME than that of the OSSO Server.

- **virtualhost** indicates that the partner application URL is a virtual host. Do not use this parameter if you are not using a virtual host.

  If you are registering a partner application URL tied to a virtual host, you must define the virtual host in httpd.conf. See “Optional: Defining Virtual Host” on page 54.

- **config_file** indicates the path where osso.conf file is to be generated.

### Optional: Defining Virtual Host

If you used a virtual host URL while registering the partner application, you must define the virtual host by updating httpd.conf on the Oracle HTTP Server that is used as the EPM System web server.

To define a virtual host:

1. **Using a text editor**, open `EPM ORACLE_INSTANCE/httpConfig/ohs/config/OHS/ohs_component/httpd.conf`.

2. Add a definition similar to the following. This definition assumes that the web server is running on the virtual server epm.myCompany.com at port epm.myCompany.com:19400. Modify the settings to suit your requirements.

   NameVirtualHost epm.myCompany.com:19400
   Listen 19400
   <VirtualHost epm.myCompany.com:19400>
   DocumentRoot "C:/Oracle/Middleware/user_projects/epmsystem1/httpConfig/ohs/config/OHS/ohs_component/private-docs"
   include "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/${COMPONENT_NAME}/mod_osso.conf"
   </VirtualHost>

### Creating mod_osso.conf

Create `mod_osso.conf` on the Oracle HTTP Server that front-ends the EPM System web server.

To create `mod_osso.conf`:

1. **Using a text editor**, create a file.

2. Copy the following content into the file and modify it for your environment.

   LoadModule osso_module C:/Oracle/Middleware/ohs/ohs/modules/mod_osso.so
   <IfModule mod_osso.c>
   OssoIpCheck off
   OssoIdleTimeout off
   OssoSecureCookies off
   </IfModule>
3 Within the `<IfModule mod_osso.c` definition, include location definitions similar to the following to identify each resource that you plan to protect using OSSO.

```html
<Location /interop/>
  require valid user
  AuthType Osso
</Location>
</IfModule>
```

4 Save the file as `mod_osso.conf`.

## Relocating `osso.conf`

The process of registering EPM System web server as a partner application (see “Registering EPM System Web Server as a Partner Application” on page 53) creates an obfuscated `osso.conf` in the location identified by the `-config_file` directive.

To relocate `osso.conf`:

1 Locate the `osso.conf` that was created when you registered EPM System web server as a partner application (see “Registering EPM System Web Server as a Partner Application” on page 53.

2 Copy `osso.conf` into the directory (on Oracle HTTP Server that front-ends the OSSO Server) identified by the `OssoConfigFile` property defined in `mod_osso.conf` (see “Creating `mod_osso.conf`” on page 54).

## Adding Cache Management Configuration for Reporting and Analysis

Edit `httpd.conf` of Oracle HTTP Server and add cache management configuration settings for Reporting and Analysis.

To add cache management configuration settings:

1 Using a text editor, open `EPM_ORACLE_INSTANCE/httpConfig/ohs/config/OHS/ohs_component/httpd.conf`.

2 Append the following directives for Reporting and Analysis cache management:

```html
<Location /WebAnalysis/>
  OssoSendCacheHeaders off
</Location>
<Location /workspace/>
  OssoSendCacheHeaders off
</Location>
<Location /hr/>
  OssoSendCacheHeaders off
</Location>
<Location /HReports/>
  OssoSendCacheHeaders off
</Location>
```

3 Save and close `httpd.conf`. 
Configuring EPM System for OSSO

Configure the OID that is integrated with the OSSO solution as an external user directory in EPM System, and then enable SSO.

To configure EPM System for OSSO:

1. Configure the OID that the OSSO solution uses as an external user directory. See "Configuring OID, Active Directory, and Other LDAP-Based User Directories" in the Oracle Enterprise Performance Management System User and Role Security Guide.

2. Enable SSO in the EPM System. “Configuring the EPM System for SSO” on page 74

   Note: To configure OSSO as the identity management solution, you must choose Other in SSO Provider or Agent, Custom HTTP Header in SSO Mechanism, and enter Proxy-Remote-User as the name of the custom HTTP header.

3. Provision at least one OID user as Shared Services administrator.

4. Restart EPM System products and custom applications that use the Shared Services security APIs.

   Note: Ensure that the OID configured with Shared Services is running before starting EPM System products.

Optional: Enabling Debug Messages on OSSO Server

To record debug messages on OSSO server, modify policy.properties. Debug messages are written to ORACLE_HOME/sso/log/ssoServer.log.

To record debug messages:

1. Using a text editor, open ORACLE_HOME/sso/conf/policy.properties; for example, C: \OraHome_1\sso\conf\policy.properties, on the OSSO server.

2. Set the value of debugLevel property to DEBUG.

   debugLevel = DEBUG

3. Save and close policy.properties.

Optional: Enabling Debug Messages for Protected Resources

To record OSSO debug messages for resources protected using mod_osso.conf, modify httpd.conf on the EPM System web server. Debug messages are written to EPM_ORACLE_INSTANCE/httpConfig/ohs/diagnostics/logs/OHS/ohs_component/ohs_component.log.

To record debug messages for protected resources:

1. Using a text editor, open EPM_ORACLE_INSTANCE/httpConfig/ohs/config/OHS/ohs_component/httpd.conf.
Set the value of `OraLogSeverity` property to `TRACE`.

```
OraLogSeverity TRACE:32
```

Save and close `httpd.conf`.

## Protecting EPM System Products for SSO

### Subtopics

- Resources to Protect
- Resources to Unprotect

You must protect EPM System resources so that SSO requests from users are redirected to the security agent (OAM, OSSO, or SiteMinder).

Oracle HTTP Server uses `mod_osso` to redirect users to the OSSO server. Users are redirected only if the URLs that they request are configured in `mod_osso` to be protected. See Managing Security in the *Oracle HTTP Server Administrator's Guide*.

For information on protecting resources for SiteMinder SSO, see SiteMinder documentation.

### Resources to Protect

Table 4 lists the contexts that must be protected. The syntax for protecting a resource (using `interop` as an example) for OSSO:

```
<Location /interop>
Require valid-user
AuthType Basic
order deny,allow
deny from all
allow from myServer.myCompany.com
satisfy any
</Location>
```

The `allow from` parameter specifies servers from which the protection of the context can be bypassed.

For EPM Workspace, Financial Reporting, and Web Analysis, you need to set only the parameters indicated in the following example:

```
<Location /workspace>
Require valid-user
AuthType Basic
</Location>
```

<table>
<thead>
<tr>
<th>EPM System Product</th>
<th>Context to Protect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Services</td>
<td>/interop</td>
</tr>
<tr>
<td>EPM System Product</td>
<td>Context to Protect</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>Oracle Hyperion Reporting and Analysis Framework</td>
<td>• /raframework</td>
</tr>
<tr>
<td></td>
<td>• /biplus_webservices</td>
</tr>
<tr>
<td>EPM Workspace</td>
<td>/workspace</td>
</tr>
<tr>
<td>Financial Reporting</td>
<td>/hr</td>
</tr>
<tr>
<td>Web Analysis</td>
<td>/WebAnalysis</td>
</tr>
<tr>
<td>Oracle Hyperion EPM Architect</td>
<td>/awb</td>
</tr>
<tr>
<td>Planning</td>
<td>/HyperionPlanning</td>
</tr>
<tr>
<td>Oracle Hyperion Performance Scorecard</td>
<td>• /HPSWebReports</td>
</tr>
<tr>
<td></td>
<td>• /HPSAlerter</td>
</tr>
<tr>
<td>Oracle Hyperion Strategic Finance</td>
<td>/HSFWebServices</td>
</tr>
<tr>
<td>Oracle Integrated Operational Planning</td>
<td>/interlace</td>
</tr>
<tr>
<td>Financial Management</td>
<td>• /hfmadf</td>
</tr>
<tr>
<td></td>
<td>• /hfmofficeprovider</td>
</tr>
<tr>
<td></td>
<td>• /hfmsmartviewprovider</td>
</tr>
<tr>
<td>Data Relationship Management</td>
<td>/dmr-web-client</td>
</tr>
<tr>
<td>Administration Services</td>
<td>/hbrlauncher</td>
</tr>
<tr>
<td>Oracle Hyperion Financial Data Quality Management</td>
<td>/HyperionFDM</td>
</tr>
<tr>
<td>Oracle Hyperion Calculation Manager</td>
<td>/calcmgr</td>
</tr>
<tr>
<td>Oracle Hyperion Provider Services</td>
<td>/aps</td>
</tr>
<tr>
<td>Oracle Hyperion Profitability and Cost Management</td>
<td>/profitability</td>
</tr>
<tr>
<td>Account Reconciliation Manager</td>
<td>/arm</td>
</tr>
<tr>
<td>Oracle Hyperion Financial Close Management</td>
<td>/fcc</td>
</tr>
<tr>
<td>Oracle Hyperion Disclosure Management†</td>
<td>/mappingtool</td>
</tr>
<tr>
<td>Oracle Hyperion Financial Data Quality Management ERP Integration Adapter for Oracle Applications</td>
<td>/aif</td>
</tr>
</tbody>
</table>

†Full certificates chain (starting from root certificate) is required on the client machine to support the use of Disclosure Management client with SSL protected web services.

**Resources to Unprotect**

Table 5 lists the contexts that must be unprotected. The syntax for unprotecting a resource (using `/interop/framework(.* )` as an example) for OSSO:

```xml
<LocationMatch /interop/framework(.* )>
  Require valid-user
</LocationMatch>
```
### Table 5  EPM System Resources to Unprotect

<table>
<thead>
<tr>
<th>EPM System Product</th>
<th>Contexts to Unprotect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Services</td>
<td>• /interop/framework(.<em>)&lt;br&gt;• /interop/Audit(.</em>)&lt;br&gt;• /interop/taskflow*&lt;br&gt;• /interop/WorkflowEngine/*&lt;br&gt;• /interop/TaskReceiver&lt;br&gt;• /framework/lcm/HSSMigration</td>
</tr>
<tr>
<td>Performance Management Architect³</td>
<td>• /awb/ces.executeAction.do&lt;br&gt;• /awb/lcm.executeAction.do&lt;br&gt;• /awb/appmanager.deployStatusUpdate.do&lt;br&gt;• /awb/jobtask.updateJobStatus.do&lt;br&gt;• /awb/integration.updateApplicationDeployStatus.do/**&lt;br&gt;• /hyperion-bpma-server</td>
</tr>
<tr>
<td>EPM Workspace</td>
<td>/workspace/browse/listXML*</td>
</tr>
<tr>
<td>Oracle Hyperion Reporting and Analysis Framework</td>
<td>• /raframework/browse/listXML&lt;br&gt;• /raframework/wsrp4j(.<em>)&lt;br&gt;• /raframework/ResourceProxy(.</em>))</td>
</tr>
<tr>
<td>Oracle Hyperion Web Analysis*</td>
<td>• /WebAnalysis/wsrp4j(.<em>))&lt;br&gt;• /WebAnalysis/ResourceProxy(.</em>))</td>
</tr>
<tr>
<td>Oracle Hyperion Financial Reporting*</td>
<td>• /hr/common/HRLogon.jsp&lt;br&gt;• /hr/wsrp4j(.<em>))&lt;br&gt;• /hr/ResourceProxy(.</em>))&lt;br&gt;• /hr/services/*&lt;br&gt;• /hr/modules/com/hyperion/reporting/web/reportViewer/HRStaticReport.jsp</td>
</tr>
<tr>
<td>Oracle Hyperion Data Relationship Management</td>
<td>/drm-migration-client</td>
</tr>
<tr>
<td>Oracle Hyperion Calculation Manager</td>
<td>/calcmgr/common.performAction.do (for Performance Management Architect)</td>
</tr>
<tr>
<td>Oracle Essbase Administration Services</td>
<td>• /eas&lt;br&gt;• /easconsole&lt;br&gt;• /easdocs</td>
</tr>
<tr>
<td>EPM System Product</td>
<td>Contexts to Unprotect</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Financial Management</td>
<td>● /hfm/EIE/EIEListener.asp</td>
</tr>
<tr>
<td></td>
<td>● /hfmapplicationservice</td>
</tr>
<tr>
<td></td>
<td>● /oracle-epm-fm-webservices</td>
</tr>
<tr>
<td></td>
<td>● /hfmlcmservice</td>
</tr>
<tr>
<td>Planning</td>
<td>● /HyperionPlanning/servlet/HspLCMServlet</td>
</tr>
<tr>
<td></td>
<td>● /HyperionPlanning/servlet/HspAppManagerServlet (for Performance Management Architect)</td>
</tr>
<tr>
<td></td>
<td>● /HyperionPlanning/Smartview</td>
</tr>
<tr>
<td></td>
<td>● /HyperionPlanning/servlet/PlanningDMEAdapter</td>
</tr>
<tr>
<td>Oracle Hyperion Performance Scorecard</td>
<td>● /HPSWebReports/wsrp4j(.*+)</td>
</tr>
<tr>
<td></td>
<td>● /HPSWebReports/ResourceProxy(.*+)</td>
</tr>
<tr>
<td></td>
<td>● /HPSWebReports/action/lcmCallback</td>
</tr>
<tr>
<td>Performance Management Architect</td>
<td>/DataSync/services*</td>
</tr>
<tr>
<td>Data Synchronization</td>
<td></td>
</tr>
<tr>
<td>Oracle Hyperion Strategic Finance</td>
<td>● /HSFWebServices/HSFWebService.asmx</td>
</tr>
<tr>
<td></td>
<td>● /HSFWebServices/HSFEntityWebService.asmx</td>
</tr>
<tr>
<td>Oracle Integrated Operational Planning</td>
<td>● /interlace/services/(.*)</td>
</tr>
<tr>
<td></td>
<td>● /interlace/anteros/(.*)</td>
</tr>
<tr>
<td></td>
<td>● /interlace/interlace/(.*)</td>
</tr>
<tr>
<td></td>
<td>● /interlace/WebHelp/(.*)</td>
</tr>
<tr>
<td></td>
<td>● /interlace/html/(.*)</td>
</tr>
<tr>
<td></td>
<td>● /interlace/email-book/(.*)</td>
</tr>
<tr>
<td>Profitability and Cost Management</td>
<td>● /profitability/cesagent</td>
</tr>
<tr>
<td></td>
<td>● /profitability/lcm</td>
</tr>
<tr>
<td></td>
<td>● /profitability/control</td>
</tr>
<tr>
<td></td>
<td>● /profitability/ApplicationListener</td>
</tr>
<tr>
<td>Oracle Hyperion Financial Data Quality Management</td>
<td>● /aif/services/FDMRuleService</td>
</tr>
<tr>
<td>ERP Integration Adapter for Oracle Applications</td>
<td>● /aif/services/RuleService</td>
</tr>
<tr>
<td>Oracle Hyperion Disclosure Management</td>
<td>● /discmanwebservices</td>
</tr>
<tr>
<td></td>
<td>● /mappingtool/MappingToolWS</td>
</tr>
</tbody>
</table>

1 /awb/integration.updateApplicationDeployStatus.do/** should be unprotected for Oracle Access Manager integration only.
SiteMinder SSO

Subtopics

- Process Flow
- Special Considerations
- Prerequisites
- Enabling SiteMinder Web Agent
- Configuring the SiteMinder Policy Server
- Configuring SiteMinder Web Server to Forward Requests to the EPM System Web Server
- Enabling SiteMinder in EPM System

SiteMinder is a Web-only solution. Desktop applications and their add-ins (for example, Microsoft Excel and Report Designer) cannot use authentication through SiteMinder. However, Oracle Hyperion Smart View for Office can use SiteMinder authentication.

Process Flow

Illustrated overview of SiteMinder-enabled SSO:

The SiteMinder SSO process:

1. Users try to access a SiteMinder protected EPM System resource. They use a URL that connects them to the web server that front-ends the SiteMinder policy server; for example,

2. The web server redirects users to the policy server, which challenges users for credentials. After verifying credentials against configured user directories, the policy server passes the credentials to the web server that hosts the SiteMinder Web Agent.

3. The web server that hosts the SiteMinder Web Agent redirects the request to the Oracle HTTP Server that front-ends EPM System. Oracle HTTP Server redirects users to the requested application deployed on WebLogic Server or IIS Server.

4. The EPM System component checks provisioning information and serves up content. For this process to work, the user directories that SiteMinder uses to authenticate users must be configured as external user directories in the EPM System. These directories must be configured as trusted.

**Special Considerations**

SiteMinder is a Web-only solution. Desktop applications and their add-ins (for example, Microsoft Excel and Report Designer) cannot use authentication through SiteMinder. However, Smart View can use SiteMinder authentication.

**Prerequisites**

1. A fully functional SiteMinder installation comprising the following components:
   - SiteMinder Policy Server on which policies and agent objects have been defined
   - SiteMinder Web Agent installed on the web server that front-ends the SiteMinder Policy Server

   When you configure the web server for EPM System components, EPM System Configurator configures the following on the Oracle HTTP Server to proxy requests to the application server:
   - `mod_wl_ohs.conf` to proxy requests to WebLogic Server
   - `mod_proxy` to proxy requests to IIS

**Enabling SiteMinder Web Agent**

The web agent is installed on a web server that intercepts requests for EPM System resources. Attempts by unauthenticated users to access a protected EPM System resources forces the web agent to challenge users for SSO credentials. When a user is authenticated, the policy server adds the login name of the authenticated user, which is carried by the header. Thereafter, the HTTP request is passed to the EPM System web server, which redirects the requests. EPM System components extracts the authenticated user credentials from headers.
SiteMinder supports SSO across EPM System products running on heterogeneous web server platforms. If EPM System products use different web servers, you must ensure that the SiteMinder cookie can be passed among web servers within the same domain. You do so by specifying the appropriate EPM System application domain as the value of the `Cookiedomain` property in the `WebAgent.conf` file of each web server.

See the “Configuring Web Agents” in the Netegrity SiteMinder Agent Guide.

**Note:** Because Shared Services uses basic authentication to protect its content, the web server that intercepts requests to Shared Services should enable basic authentication to support SSO with SiteMinder.

You configure the web agent by running the SiteMinder Web Agent Configuration wizard (by executing `WEBAGENT_HOME/install_config_info/nete-wa-config`; for example, `C:\netegrity\webagent\install_config_info\nete-wa-config.exe` on Windows). The configuration process creates a `WebAgent.conf` for the SiteMinder web server.

### To enable SiteMinder Web Agent:

1. **Using a text editor, open `WebAgent.conf`.** The location of this file depends on the web server that you are using. If you are configuring the an IIS Server as the SiteMinder web server, the location of `WebAgent.conf` is `WEB_AGENT_HOME/bin/IIS`; for example, `C:\SiteMinder\webagent\bin\iis\WebAgent.conf`.

2. **Set the value of `enableWebAgent` property to Yes.**
   ```
   enableWebAgent="YES"
   ```

3. **Save and close the web agent configuration file.**

### Configuring the SiteMinder Policy Server

A SiteMinder administrator must configure the policy server to enable SSO to EPM System products.

The configuration process involves:

- Creating a SiteMinder Web Agent and adding configuration objects appropriate for the SiteMinder web server
- Creating a realm for each EPM System resource that should be protected and adding the web agent to the realm. See “Resources to Protect” on page 57
- Within the realm that was created for protected EPM System resources, create realms for unprotected resources. See “Resources to Unprotect” on page 58
- Creating HTTP header reference. The header should provide the value of `Login Attribute` to EPM System applications. See “Configuring OID, Active Directory, and Other LDAP-Based User Directories” in the Oracle Enterprise Performance Management System User and Role Security Guide for a brief description of `Login Attribute`.
- Creating rules within the realms with Get, Post, and Put as web agent actions
Creating a response attribute with `hyplogin=<%userattr="SM_USERLOGINNAME"%>` as the value

Creating a policy, assigning user directory access, and adding rules that you created for EPM System to Current Members list

Setting responses for the rules you created for EPM System components

**Configuring SiteMinder Web Server to Forward Requests to the EPM System Web Server**

Configure the web server that hosts the SiteMinder web agent to forward requests from authenticated users (containing the header identifying the user) to the EPM System web server.

For Apache-based web servers, use directives similar to the following to forward authenticated requests:

```
ProxyPass / http://EPM_WEB_SERVER:EPM_WEB_SERVER_PORT/
ProxyPassReverse / http://EPM_WEB_SERVER:EPM_WEB_SERVER_PORT/
ProxyPreserveHost On
#If SiteMinder Web Server is using HTTPS but EPM Web Server is using HTTP
RequestHeader set WL-Proxy-SSL true
```

In this directive, replace `EPM_WEB_SERVER` and `EPM_WEB_SERVER_PORT` with the actual values for your environment.

**Enabling SiteMinder in EPM System**

Integration with SiteMinder requires that you enable SiteMinder authentication for EPM System products. See “Configuring the EPM System for SSO” on page 74.

**Kerberos Single Sign-on**

**Subtopics**

- Overview
- Support Limitations
- Assumptions: Kerberos Environment
- Kerberos SSO with WebLogic Server
- WebLogic Server Procedures to Support Kerberos Authentication

**Overview**

EPM System products support Kerberos SSO if the application server that hosts EPM System products is set up for Kerberos authentication.

Kerberos is a trusted authentication service, where each Kerberos client trusts the identities of other Kerberos clients (users, network services, and so on) to be valid.
The following steps list what happens when a user accesses an EPM System product:

- From a Windows computer, the user logs in to a Kerberos realm.
- Using a browser that is configured to use Integrated Windows Authentication, the user tries to log into EPM System products running on the application server.
- The application server (Negotiate Identity Asserter) intercepts the request and gets the Simple and Protected Generic Security Services API (GSSAPI) Negotiation Mechanism (SPNEGO) token with the Kerberos ticket from the browser’s authorization header.
- The asserter validates the user’s identity included in the token against its identity store to pass information about the user to EPM System product. The EPM System product validates the user name against an Active Directory. The EPM System product issues an SSO token that supports SSO across all EPM System products.

**Support Limitations**

Kerberos SSO is supported for all EPM System products, with the following exceptions:

- Kerberos SSO is not supported for thick clients including Smart View.
- Kerberos SSO support for IIS-embedded EPM System products (for example, Financial Management) is available only through EPM Workspace. SSO access to Oracle Hyperion Financial Data Quality Management, is provided through Financial Management.

**Assumptions: Kerberos Environment**

This document assumes the following:

- A fully functional Kerberos-enabled network environment
  - The corporate Active Directory is configured for Kerberos authentication.
  - The application server and HTTP server machines that host EPM System products are within the Kerberos realm.
  - The machines from which EPM System products are accessed are part of the Kerberos realm.
  - Browsers used to access EPM System products are configured for Integrated Windows Authentication. For information on enabling Integrated Windows Authentications, see:
    - Internet Explorer documentation on the Microsoft Help and Support web site.
    - Firefox documentation on the Firefox Support web site.
- EPM System product users have Kerberos credentials that enable them to log in to client machines in the domain.
- Integrated Windows Authentication is disabled in IIS if it is used as the web server for EPM System products.
Kerberos SSO with WebLogic Server

WebLogic Server Kerberos SSO uses the Negotiate Identity Asserter to negotiate and decode SPNEGO tokens to enable SSO with Microsoft clients. WebLogic Server decodes SPNEGO tokens to obtain Kerberos ticket and validates and maps the ticket to a WebLogic Server user. You can use the Active Directory Authenticator of WebLogic Server with the Negotiate Identity Asserter to configure Active Directory as the user directory for WebLogic Server users.

When the browser requests access to an EPM System product, KDC issues a Kerberos ticket to the browser, which creates a SPNEGO token containing the supported GSS token types. The Negotiate Identity Asserter decodes the SPNEGO token and uses GSSAPIs to accept the security context. The identity of the user who initiated the request is mapped to a user name and passed back to WebLogic Server. Additionally, the WebLogic Server determines the groups to which the user belongs. At this stage, the requested EPM System product is made available to the user.

**Note:** The user must use a browser that supports the SPNEGO (for example, Internet Explorer or Firefox) to access the EPM System products running on WebLogic Server. WebLogic Server can run on a UNIX or Windows platform.

Using the user ID derived from the authentication process, the EPM System product authorization process checks for provisioning data. Access to EPM System product is restricted based on provisioning data.

This document is based on the following assumptions. See “Assumptions: Kerberos Environment” on page 65 for assumptions related to the network environment.

- Active Directory security groups and users are available to support the WebLogic Server to Active Directory handshake. See “Configuring Single Sign-on with Microsoft Clients” in Oracle Fusion Middleware Securing Oracle WebLogic Server.

  The Active Directory user must be able to log in to WebLogic Server as a power user, preferably as WebLogic Server Administrator. The user account is updated by selecting Use DES encryption types for this account.

  See Microsoft documentation for detailed information.

  The configuration must support the use of the web server DNS name (reverse proxy) as Kerberos Service Principal Name.

- The myrealm security realm in the WebLogic Server domain is modified to add Active Directory as the authentication provider. See WebLogic Server documentation for detailed information.

WebLogic Server Procedures to Support Kerberos Authentication

A WebLogic Server administrator should complete these tasks to support Kerberos authentication:
Configure the WebLogic domain of EPM System. See “Configuring EPM System WebLogic Domain” on page 67.

Create an authentication provider. See “Creating an LDAP Authentication Provider in WebLogic Server” on page 67.

Create a Negotiate Identity Asserter. See “Creating a Negotiate Identity Asserter” on page 68.

Create a Kerberos identification. See “Creating Kerberos Identification for WebLogic Server” on page 68.

Create a Kerberos configuration file. See “Creating Kerberos Configuration File” on page 68.

Update WebLogic startup script. See “Updating WebLogic Startup Script” on page 69.


Deploy and use SSODiag to verify that the WebLogic Server is ready to support Kerberos SSO for EPM System. See “Using SSODiag to Test the Kerberos Environment” on page 70.

---

## Configuring EPM System WebLogic Domain

Generally, EPM System products are deployed into epmsysteml, which is the default WebLogic domain. This domain is identified also as EPM_ORACLE_INSTANCE.

To configure the EPM System WebLogic domain for Kerberos authentication:

1. Install EPM System components.
2. Create the WebLogic domain by configuring and deploying Foundation Services only.

Steps involved:

- “Creating an LDAP Authentication Provider in WebLogic Server” on page 67
- “Creating a Negotiate Identity Asserter” on page 68
- “Creating Kerberos Identification for WebLogic Server” on page 68
- “Updating WebLogic Startup Script” on page 69
- “Configuring Authorization Policies” on page 69

---

## Creating an LDAP Authentication Provider in WebLogic Server

An LDAP Authentication provider stores user and group information in an external LDAP server. LDAP v2- or v3- compliant LDAP server should work with WebLogic Server. See Configuring LDAP authentication providers in Oracle Fusion Middleware Securing Oracle WebLogic Server guide.
Creating a Negotiate Identity Asserter

The Negotiate Identity Assertion provider enables SSO with Microsoft clients. It decodes SPNEGO tokens to obtain Kerberos tokens, validates the Kerberos tokens, and maps the tokens to WebLogic users. The Negotiate Identity Assertion provider, an implementation of the Security Service Provider Interface (SSPI) as defined by the WebLogic Security Framework, provides the necessary logic to authenticate a client based on the client's SPNEGO token. See Configuring negotiate identity assertion provider in the Oracle Fusion Middleware Securing Oracle WebLogic Server guide.

While creating the Negotiate Identity Assertion provider, set the JAAS Control Flag option to OPTIONAL for all Authenticators. See “Set the JAAS control flag” in Oracle Fusion Middleware Oracle WebLogic Server Administration Console Online Help.

Creating Kerberos Identification for WebLogic Server

Create Active Directory user objects that represent WebLogic Server and EPM System web server and map them to service principal names (SPN). SPNs are unique identifiers that identify the service to clients on the network.

To create Kerberos identification for WebLogic Server:

1. Create an Active Directory user that complies with the Kerberos protocol. The user account's encryption type must be DES. See Creating Kerberos identification for WebLogic Server in the Oracle Fusion Middleware Securing Oracle WebLogic Server guide.

For example, Active Directory user wls-myServer0055 may represent the WebLogic Server running on computer myServer0055.

- While creating the user, do not select password options.
- After creating the user, modify the user properties and select Use DES encryption types for this account.
- Reset the password of the user account.

2. Use the setspn command similar to the following to map the Kerberos SPN, HTTP/WEBLOGIS_SERVER_HOST_NAME to a Microsoft user account.

```bash
setspn -A HTTP/myServer0055.myexample.com wls-myServer0055
```

3. Create a Kerberos keytab file using a command such as the following and make it available to WebLogic Server:

```bash
ktpass -out c:\temp\wls-myServer0055.keytab -princ HTTP/myServer0055.myexample.com@EXAMPLE.COM -mapuser wls-myExample0055 -pass PASSWORD -DesOnly
```

Creating Kerberos Configuration File

Kerberos configuration properties are defined in kerb.ini. This configuration file is required to use Kerberos administration tools such as kinit and ktab.
See Configuring Your Network Domain to Use Kerberos in Oracle Fusion Middleware Securing Oracle WebLogic Server 11g Release 1 (10.3.1).

### Updating WebLogic Startup Script

See Using Startup Arguments for Kerberos Authentication with WebLogic Server and Creating a JAAS Login File in Oracle Fusion Middleware Securing Oracle WebLogic Server 11g Release 1 (10.3.1).

If EPM System managed servers are run as Windows services, update the Windows registry to set the JVM startup options.

1. To update JVM Startup options in Windows registry:
   1. Open Windows Registry Editor.
   2. Find the Foundation Services key by selecting My Computer, then HKEY_LOCAL_MACHINE, then Software, then Oracle, and then Foundation Services.
   3. Add the following string values:

   **Note:** The names listed in Table 6 are examples.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>JVMOption14</td>
<td>REG_SZ</td>
<td>-Djava.security.krb5.kdc=Active Directory host name or IP address</td>
</tr>
<tr>
<td>JVMOption15</td>
<td>REG_SZ</td>
<td>-Djava.security.auth.login.config=krb5Login.conf</td>
</tr>
<tr>
<td>JVMOption16</td>
<td>REG_SZ</td>
<td>-Djavax.security.auth.useSubjectCredsOnly=false</td>
</tr>
<tr>
<td>JVMOption17</td>
<td>REG_SZ</td>
<td>-Djava.security.enableNegotiate=true</td>
</tr>
</tbody>
</table>

### Configuring Authorization Policies

See Options for Securing Web Application and EJB Resources in the Oracle Fusion Middleware Securing Resources Using Roles and Policies for Oracle WebLogic Server guide for information on configuring authorization policies for Active Directory users who access the EPM System.

For sample policy configuration steps, see “Creating Policies for SSODiag” on page 71.
### Using SSODiag to Test the Kerberos Environment

#### Subtopics
- Deploying SSODiag
- Configuring Oracle HTTP Server for SSODiag
- Creating Policies for SSODiag
- Using SSODiag to Test WebLogic Server Configuration for Kerberos Authentication

SSODiag is a diagnostic web application that tests whether WebLogic Server in your Kerberos environment is ready to support EPM System.

#### Deploying SSODiag

Use the credentials (default user name is `epm_admin`) that you specified while deploying Foundation Services to deploy SSODiag.

1. To deploy and configure SSODiag:
   1. Log on to the WebLogic Server Administration Console for EPM System domain.
   2. Using the Install Application Assistant, select `EPM_ORACLE_HOME/products/Foundation/AppServer/InstallableApps/common/SSODiag.war` as the web application to install.
   3. Deploy SSODiag as an application (choose **Install this deployment as an application** as targeting style) using **CustomRolesAndPolicies** security option.
   4. Activate the changes you made.

#### Configuring Oracle HTTP Server for SSODiag

Update `mod_wl_ohs.conf` to configure Oracle HTTP Server to forward SSODiag URL requests to the WebLogic Server.

1. To configure URL forwarding in Oracle HTTP Server:
   1. Using a text editor, open `EPM_ORACLE_INSTANCE/httpConfig/ohs/config/OHS/ohs_component/mod_wl_ohs.conf`.
   2. Add a `LocationMatch` definition for SSODiag:
      ```xml
      <LocationMatch /SSODiag/>
      SetHandler weblogic-handler
      WeblogicCluster myServer:28080
      </LocationMatch>
      ```
      In the preceding sample, `myServer` denotes the Foundation Services host machine and `28080` represents the port at which Shared Services listens for requests.
   3. Save and close `mod_wl_ohs.conf`.
   4. Restart Oracle HTTP Server.
Creating Policies for SSODiag

Create a policy in the WebLogic Server Administrative Console to protect the following SSODiag URL.

http://OHS_HOST_NAME:PORT/SSODiag/krbssodiag

In this sample, OHS_HOST_NAME indicates the name of the server that hosts Oracle HTTP Server and PORT indicates the port where Oracle HTTP Server listens for requests.

To create policies to protect SSODiag:

1. In the Change Center in WebLogic Server Administration Console for EPM System domain, select Lock & Edit.
2. Select Deployments, then SSODiag, then Security, then Roles, and then URL Patterns.
3. Create the following URL patterns:
   - /index.jsp

4. Modify each URL pattern that you created:
   a. From the list of URL patterns in Stand-Alone Web Application URL Patterns, open the pattern (/) that you created by clicking it.
   b. Select Add Conditions.
   d. Select Next.
   e. In User Argument Name, enter the Active Directory user whose account is used to access a client desktop configured for Kerberos authentication; for example, krbuser1, and select Add.
   f. Select Finish.
5. Select Save.

Using SSODiag to Test WebLogic Server Configuration for Kerberos Authentication

If WebLogic Server configuration for Kerberos authentication works correctly, the Oracle Hyperion Kerberos SSO diagnostic Utility V 1.0 page displays the following message:

Retrieving Kerberos User principal name... Success.
Kerberos principal name retrieved... SOME_USER_NAME

Caution! Do not configure EPM System components for Kerberos authentication if SSODiag cannot retrieve the Kerberos principal name.

To test WebLogic Server configuration for Kerberos authentication:

1. Start Foundation Services and Oracle HTTP Server.
Using WebLogic Server Administration Console, start SSODiag web application to service all requests.

Log on to a client machine configured for Kerberos authentication using valid Active Directory credentials.

Using a browser, connect to the following SSODiag URL:

http://OHS_HOST_NAME:PORT/SSODiag/krbssodiag

In this sample, OHS_HOST_NAME indicates the name of the server that hosts Oracle HTTP Server, and PORT indicates the port where Oracle HTTP Server listens for requests.

If Kerberos authentication works properly, SSODiag displays the following information:

Retrieving Kerberos User principal name... Success.
Kerberos principal name retrieved... SOME_USER_NAME

If Kerberos authentication does not work properly, SSODiag displays the following information:

Retrieving Kerberos User principal name... failed.

Configuring Foundation Services for Kerberos Authentication

Subtopics

- Changing the Security Model
- Updating EPM System Security Configuration
- Testing Kerberos SSO

Caution! Complete this step only after SSODiag successfully retrieves the Kerberos principal name. See “Using SSODiag to Test WebLogic Server Configuration for Kerberos Authentication” on page 71.

Changing the Security Model

The default security model for web applications secured by the security realm is DDonly. You must change the security model to CustomRolesAndPolicies.

To change the security model:

1 Using a text editor, open MIDDLEWARE_HOME/user_projects/domains/EPMSystem/config/config.xml.
2 Locate the following element in the application deployment descriptor for each Foundation Services component:
   <security-dd-model>DDOnly</security-dd-model>
3 Change the security model as follows for each component:
   <security-dd-model>CustomRolesAndPolicies</security-dd-model>
4 Save and close config.xml.
Updating EPM System Security Configuration

Change EPM System security configuration to enable Kerberos SSO.

To configure EPM System for Kerberos authentication:

1. Log on to Shared Services Console as administrator.
2. Add the Active Directory domain that is configured for Kerberos authentication as an external user directory in Shared Services. See “Configuring OID, Active Directory, and Other LDAP-based User Directories” in the Oracle Enterprise Performance Management System User and Role Security Guide.

In Security Options, select the settings in Table 7 to enable Kerberos SSO.

Table 7  Settings to Enable Kerberos SSO

<table>
<thead>
<tr>
<th>Field</th>
<th>Required Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable SSO</td>
<td>Selected</td>
</tr>
<tr>
<td>SSO Provider or Agent</td>
<td>Other</td>
</tr>
<tr>
<td>SSO Mechanism</td>
<td>Get Remote User from HTTP Request</td>
</tr>
</tbody>
</table>

4. Restart Foundation Services.

Testing Kerberos SSO

Log in to Foundation Services to verify that Kerberos SSO is working properly.

To test Kerberos SSO:

1. Verify that Foundation Services and Oracle HTTP Server are running.
2. Log on to a client machine configured for Kerberos authentication using a valid Active Directory credentials.
3. Using a browser, connect to the Foundation Services URL.

Configuring Other EPM System Components

Using EPM System Configurator, configure and deploy other EPM System components into the WebLogic domain where Foundation Services is deployed.

Change the security model for each EPM System Component deployed into the WebLogic domain where Foundation Services is deployed. See “Changing the Security Model” on page 72.
Configuring the EPM System for SSO

EPM System products must be configured to support security agent for SSO. The configuration specified in Shared Services determines the following for all EPM System products:

- Whether to accept SSO from a security agent
- The authentication mechanism to accept for SSO

In an SSO-enabled environment, the EPM System product that is first accessed by the user parses the SSO mechanism to retrieve the authenticated user ID contained in it. The EPM System product checks the user ID against the user directories configured in Shared Services to determine that the user is a valid EPM System user. It also issues a token that enables SSO across EPM System products.

The configuration specified in Shared Services enables SSO and determines the authentication mechanism to accept for SSO for all EPM System products.

➤ To enable SSO from a web identity management solution:


2. Select Administration, then Configure User Directories.

3. Verify that the user directories used by the web identity management solution are configured as external user directories in Shared Services.

   For example, to enable Kerberos SSO, you must configure the Active Directory that is configured for Kerberos authentication as an external user directory.


5. Select Show Advanced Options.

6. In **Single Sign-on Configuration** in the Defined User Directories screen, perform the following steps.
   a. Select **Enable SSO**.

   b. From **SSO Provider or Agent**, select a web identity management solution. Choose **Other** if you are configuring SSO with Kerberos.

      The recommended SSO mechanism is automatically selected. See Table 8. See “Supported SSO Methods” on page 47.

**Note:** If you are not using the recommended SSO mechanism, you must choose **Other** in **SSO Provider or Agent**. For example, to use a mechanism other than HTTP Header for SiteMinder, choose **Other** in **SSO Provider or Agent** and then select the SSO Mechanism that you want to use in **SSO Mechanism**.
Table 8  Preferred SSO Mechanisms for Web Identity Management Solutions

<table>
<thead>
<tr>
<th>Web Identity Management Solution</th>
<th>Recommended SSO Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Access Manager</td>
<td>Custom HTTP Header¹</td>
</tr>
<tr>
<td>OSSO</td>
<td>Select Other in SSO Provider or Agent and Custom HTTP Header in SSO Mechanism. Enter Proxy-Remote-User as the name of the custom HTTP header.</td>
</tr>
<tr>
<td>SiteMinder</td>
<td>Custom HTTP Header</td>
</tr>
<tr>
<td>Kerberos</td>
<td>WebLogic Server: Custom HTTP Header</td>
</tr>
</tbody>
</table>

¹The default HTTP Header name is HYPLOGIN. If you are using a custom HTTP Header, replace the name.

7  Click OK.

**Single Sign-on Options for Smart View**

Although Smart View is a thick client and not a browser, it connects to server components using HTTP and behaves much like a browser from a system perspective. Smart View supports all standard web-based integration methods that browser interfaces support. However, there are some limitations:

- Smart View is not supported in Kerberos-enabled environments.
- SSO mechanisms are supported for shared connections only. SSO mechanisms are not supported with private connections, which are used primarily for backward compatibility.
- If Smart View is launched from an existing browser session that is connected to an EPM System component, users must sign into Smart View again because it does not share the cookie from the existing session.
Overview

A custom authentication module is a Java module that customers develop and implement to authenticate EPM System users. Generally, EPM System products use a logon screen to capture the user name and password, which are used to authenticate users. Instead of using EPM System authentication, you can use a custom authentication module to authenticate users and pass authenticated user credentials to EPM System for further processing. Implementing a custom authentication module does not involve modifying EPM System products.

You can use a custom authentication module with both the thick clients (for example, Oracle Hyperion Smart View for Office, and Oracle Essbase Studio) and thin clients (for example, EPM Workspace).

The custom authentication module uses the information a user enters when logging in to an EPM System product. If enabled for a user directory, it authenticates users through the custom authentication module. On successfully authenticating the user, the custom authentication module returns the user name to EPM System.

The following illustration presents a sample custom authentication scenario:
For example, you can use RSA SecurID infrastructure as the custom provider to ensure transparent strong authentication to the EPM System. An overview:

1. The user enters credentials (generally, user name and password) to access an EPM System product. These credentials should uniquely identify the user to the provider used by the custom authentication module. For example, if you are using an RSA SecurID infrastructure to authenticate users, the user enters an RSA user ID and PIN (not an EPM System user ID and password).

2. Using the search order (see “Search Order” on page 80), EPM System cycles through configured user directories to locate the user.
   - If the current user directory is not configured for custom authentication, EPM System tries to locate and authenticate the user through EPM System authentication.
   - If the user directory is configured for custom authentication, EPM System delegates the authentication process to the custom module.

3. If EPM System delegates authentication to the custom module, the custom authentication module accepts the credentials and uses its own logic to direct user authentication against a custom provider, for example, RSA SecurID infrastructure.

4. If the custom authentication module authenticates the user against its provider, it returns the user name to the EPM System, or it returns a Java exception.

   The user name returned by the custom authentication module must be identical to a user name in one of the user directories that is enabled for custom authentication.
   - If the custom authentication module returns a user name, EPM System locates the user in a user directory that is enabled for custom authentication. At this stage, EPM System does not search the user directories that are not configured for custom authentication.
• If the custom authentication module throws an exception or returns a null user, EPM System continues to search for the user in the remaining user directories in the search order that are not enabled for custom authentication. If a user who matches the credentials is not found, EPM System displays an error.

### Use-Case Examples and Limitations

Custom authentication implementation scenarios include the following:

• Adding one-time password Support
• Performing authentication against a Resource Access Control Facility (RACF)
• Adding Simple Authentication and Security Layer (SASL) bind to LDAP-enabled user directories instead of simple LDAP binds

Authentication with challenge/response mechanism may not work well if you implement a custom authentication module. Custom messages thrown by the custom authentication module are not propagated to the clients. Because clients, for example, EPM Workspace, override the error message to display a generic message, the following scenarios are not valid:

• Two consecutive RSA SecurID PINs
• Password variant with challenges, such as enter first, last, and third characters of password

### Prerequisites

• Access to Shared Services as Shared Services administrator
Design and Coding Considerations

Subtopics

- Search Order
- User Directories and Custom Authentication Module
- CSSCustomAuthenticationIF Java Interface

Search Order

In addition to Native Directory, multiple user directories can be configured in Shared Services. A default search order position is assigned to all configured user directories. You can modify the search order from Shared Services Console. Excepting Native Directory, you can remove configured user directories from the search order. EPM System does not use the user directories that are not included in the search order. See the Oracle Enterprise Performance Management System User and Role Security Guide.

The search order determines the order in which EPM System cycles through the user directories to authenticate users. If the user is authenticated in a user directory, EPM System stops the search and returns the user. EPM System denies authentication and returns an error if the user cannot be authenticated against user directories in the search order.

Impact of Custom Authentication on Search Order

Custom authentication affects how EPM System security interprets the search order.

If the custom authentication module returns a user name, EPM System locates the user only in a user directory that is enabled for custom authentication. At this stage, EPM System ignores user directories that are not configured for custom authentication.

Understanding the Custom Authentication Flow

The following use case scenarios are used to explore custom authentication flow:

- “Use Case Scenario 1” on page 80
- “Use-case Scenario 2” on page 82
- “Use-case Scenario 3” on page 82

Use Case Scenario 1

Table 9 details the EPM System user directory configuration and search order used in this scenario. This scenario assumes that the custom authentication module uses an RSA infrastructure to authenticate users.
Table 9  Setup for Scenario 1

<table>
<thead>
<tr>
<th>User Directory Type and Name</th>
<th>Search Order</th>
<th>Custom Authentication</th>
<th>Sample User Names</th>
<th>Password¹</th>
</tr>
</thead>
<tbody>
<tr>
<td>Native Directory</td>
<td>1</td>
<td>Disabled</td>
<td>test_user_1</td>
<td>password</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_user_2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_user_3</td>
<td></td>
</tr>
<tr>
<td>LDAP-Enabled SunONE_West</td>
<td>2</td>
<td>Disabled</td>
<td>test_ldap_1</td>
<td>ldappassword</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_ldap_2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_ldap_3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_ldap_4</td>
<td></td>
</tr>
<tr>
<td>LDAP-Enabled SunONE_East</td>
<td>3</td>
<td>Enabled</td>
<td>test_ldap_1</td>
<td>ldappassword on SunONE and RSA PIN in custom module</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_ldap_2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_user_3</td>
<td></td>
</tr>
</tbody>
</table>

¹For simplicity, it is assumed that all users use the same user directory password.

To initiate the authentication process, a user enters a user name and password in the logon screen of an EPM System product.

In this scenario, the custom authentication module performs the following actions:

- Accepts a user name and RSA PIN as the user credentials
- Returns a user name in **username@providername** format, for example, test_ldap_2@SunONE_East, to EPM System security.

Table 10  User interaction and results

<table>
<thead>
<tr>
<th>User Name and Password</th>
<th>Authentication Result</th>
<th>Login User Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>test_user_1/password</td>
<td>Success</td>
<td>Native Directory</td>
</tr>
<tr>
<td>test_user_3/password</td>
<td>Success</td>
<td>Native Directory</td>
</tr>
<tr>
<td>test_user_3/ldappassword</td>
<td>Success</td>
<td>SunONE_West (search order 2)¹</td>
</tr>
<tr>
<td>test_user_3/RSA PIN</td>
<td>Success</td>
<td>SunONE_East (search order 3)²</td>
</tr>
<tr>
<td>test_ldap_2/ldappassword</td>
<td>Success</td>
<td>SunONE_West (search order 2)</td>
</tr>
<tr>
<td>test_ldap_4/RSA PIN</td>
<td>Failure</td>
<td>EPM System displays an authentication error.³</td>
</tr>
</tbody>
</table>

¹The custom authentication cannot authenticate this user because the user entered EPM System credentials. EPM System can identify this user only in a user directory that is not enabled for custom authentication. The user is not in Native Directory (search order number 1) but is identified in SunONE West (search order number 2).

²EPM System does not find this user in Native Directory (search order number 1) or SunONE West (search order number 2). The custom authentication module validates the user against RSA Server and returns test_user_3@SunONE_EAST to EPM System. EPM System locates the user in SunONE East (search order number 3), which is a custom authentication–enabled user directory.

³Oracle recommends that all users authenticated by the custom module be present in a custom authentication–enabled user directory included in the search order. Login fails if the user name that is returned by the custom authentication module is not present in a custom authentication-enabled user directory included in the search order.
Use-case Scenario 2

Table 11 details the EPM System user directory configuration and search order used in this scenario. This scenario assumes that the custom authentication module uses an RSA infrastructure to authenticate users.

In this scenario, the custom authentication module performs the following actions:

- Accepts a user name and RSA PIN as the user credentials
- Returns a user name, for example, test_1dap_2, to EPM System security

<table>
<thead>
<tr>
<th>User Directory</th>
<th>Search Order</th>
<th>Custom Authentication</th>
<th>Sample User Names</th>
<th>Password¹</th>
</tr>
</thead>
<tbody>
<tr>
<td>Native Directory</td>
<td>1</td>
<td>Disabled</td>
<td>test_user_1</td>
<td>password</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_user_2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_user_3</td>
<td></td>
</tr>
<tr>
<td>LDAP-Enabled, for example, SunONE</td>
<td>2</td>
<td>Enabled</td>
<td>test_ldap1</td>
<td>ldappassword on SunONE and RSA PIN in custom module</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_ldap2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_user_3</td>
<td></td>
</tr>
</tbody>
</table>

¹For simplicity, it is assumed that all users use the same user directory password.

To initiate the authentication process, a user enters a user name and password on the login screen of an EPM System product.

<table>
<thead>
<tr>
<th>User Name and Password</th>
<th>Login Result</th>
<th>Login User Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>test_user_1/password</td>
<td>Success</td>
<td>Native Directory</td>
</tr>
<tr>
<td>test_user_3/password</td>
<td>Success</td>
<td>Native Directory</td>
</tr>
<tr>
<td>test_user_3/ldappassword</td>
<td>Failure</td>
<td>SunONE¹</td>
</tr>
<tr>
<td>test_user_3/RSA PIN</td>
<td>Success</td>
<td>SunONE²</td>
</tr>
</tbody>
</table>

¹Authentication of user against Native Directory fails because of password mismatch. Authentication of user using the custom authentication module fails because the password used is not a valid RSA PIN. EPM System does not try to authenticate this user in SunONE (search order 2), because custom authentication settings override EPM System authentication in this directory.

²Authentication of user against Native Directory fails because of password mismatch. The custom authentication module authenticates the user and returns the user name test_user_3 to EPM System.

Use-case Scenario 3

Table 13 details the EPM System user directory configuration and search order used in this scenario. This scenario assumes that the custom authentication module uses an RSA infrastructure to authenticate users.

For clarity in such scenarios, Oracle recommends that your custom authentication module return the user name in username@providername format; for example, test_ldap_4@SunONE.
Table 13  A sample search order

<table>
<thead>
<tr>
<th>User Directory</th>
<th>Search Order</th>
<th>Custom Authentication</th>
<th>Sample User Names</th>
<th>Password¹</th>
</tr>
</thead>
<tbody>
<tr>
<td>Native Directory</td>
<td>1</td>
<td>Enabled</td>
<td>test_user_1</td>
<td>RSA_PIN</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_user_2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_user_3</td>
<td></td>
</tr>
<tr>
<td>LDAP-Enabled, for example, MSAD</td>
<td>2</td>
<td>Disabled</td>
<td>test_ldap1</td>
<td>ldappassword</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_ldap4</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_user_3</td>
<td></td>
</tr>
<tr>
<td>LDAP-Enabled, for example, SunONE</td>
<td>3</td>
<td>Enabled</td>
<td>test_ldap1</td>
<td>ldappassword on SunONE and RSA PIN in custom module</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_ldap4</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>test_user_3</td>
<td></td>
</tr>
</tbody>
</table>

¹For simplicity, it is assumed that all users use the same user directory password.

To initiate the authentication process, a user enters a user name and password in the logon screen of an EPM System product.

Table 14  User interaction and results

<table>
<thead>
<tr>
<th>User Name and Password</th>
<th>Authentication Result</th>
<th>Login User Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>test_user_1/password</td>
<td>Success</td>
<td>Native Directory</td>
</tr>
<tr>
<td>test_user_3/RSA_PIN</td>
<td>Success</td>
<td>Native Directory</td>
</tr>
<tr>
<td>test_user_3/ldappassword</td>
<td>Success</td>
<td>MSAD (search order 2)</td>
</tr>
<tr>
<td>test_ldap_4/ldappassword</td>
<td>Success</td>
<td>MSAD (search order 2)</td>
</tr>
<tr>
<td>test_ldap_4/RSA PIN</td>
<td>Success</td>
<td>SunONE (search order 3)</td>
</tr>
</tbody>
</table>

User Directories and Custom Authentication Module

To use the custom authentication module, user directories that contain EPM System user and group information can be individually configured to delegate authentication to the custom module.

EPM System users who are authenticated using a custom module must be present in one of the user directories included in the search order (see “Search Order” on page 80). Also, the user directory must be configured to delegate authentication to the custom module.

The identity of the user in the custom provider (for example, 1357642 in RSA SecurID infrastructure) may be different from the user name in the user directory (for example, jDoe in an Oracle Internet Directory) configured in Shared Services. After authenticating the user, the custom authentication module must return the user name jDoe to EPM System.
Note: As a best practice, Oracle recommends that the user name in the user directories configured in EPM System be identical to those available on the user directory used by the custom authentication module.

**CSSCustomAuthenticationIF Java Interface**

The custom authentication module must use the CSSCustomAuthenticationIF Java interface to integrate with EPM System security framework. It must return a user name string if custom authentication is successful or an error message if authentication is not successful. For the authentication process to be completed, the user name returned by the custom authentication module must be present in one of the user directories included in Shared Services search order. EPM System security framework supports the `username@providerName` format.

Note: Ensure that the user name that the custom authentication module returns does not contain an * (asterisk), because EPM System security framework interprets it as a wildcard character while searching for users.

See “Sample Code 1” on page 95 for CSSCustomAuthenticationIF interface signature.

Your custom authentication module can be a class file must be included in `CustomAuth.jar`. The package structure is unimportant.

For detailed information about the CSSCustomAuthenticationIF interface, see Security API documentation.

The `authenticate` method of CSSCustomAuthenticationIF supports custom authentication. The `authenticate` method accepts credentials (user name and password) that the user entered while trying to access the EPM System as input parameters. This method returns a string (user name) if custom authentication is successful. It throws a `java.lang.Exception` if authentication is unsuccessful. The user name returned by the method should uniquely identify a user in one of the user directories included in Shared Services search order. EPM System security framework supports the `username@providerName` format.

Note: To initialize resources, for example, a JDBC connection pool use the class constructor. Doing so improves performance by not loading resources for every authentication.

**Deploying the Custom Authentication Module**

Subtopics

- Overview of Steps
- Updating Settings in Shared Services
- Testing Your Deployment

Only one custom module is supported for an EPM System deployment. You can enable custom authentication for one or more user directories in the search order.
The custom authentication module must implement the public interface
CSSCustomAuthenticationIF, defined in the com.hyperion.css package. This document
assumes that you have a fully functional custom module that defines the logic for authenticating
users against the user provider of your choice. After you develop and test a custom authentication
module, you must implement it in EPM System environment.

Overview of Steps

**Note:** Your custom authentication code should not use log4j for error logging. If the code that
you used in a previous release uses log4j, you must remove it from the code before using
it with this release.

To implement the custom authentication module, complete the following steps:

- Stop EPM System products including Shared Services and any systems that use Shared
  Services APIs.
- Copy CustomAuth.jar into EPM_ORACLE_HOME/user_projects/domains/
  WEBLOGIC_DOMAIN/lib, typically, Oracle/Middleware/user_projects/domains/
  EPMSystem/lib.
  
  **Note:** If you are upgrading from Release 11.1.2.0 or 11.1.2.1 that had an implementation of
custom authentication module, move CustomAuth.jar from EPM_ORACLE_HOME/
  common/jlib/11.1.2.0 into EPM_ORACLE_HOME/user_projects/domains/
  WEBLOGIC_DOMAIN/lib.

- Update user directory settings in Shared Services. See “Updating Settings in Shared Services”
on page 85.
- Start Shared Services followed by other EPM System products.
- Test your implementation. See “Testing Your Deployment” on page 86.

Updating Settings in Shared Services

**Subtopics**

- Updating User Directory Configurations
- Updating Security Options

By default, custom authentication is disabled for all user directories. You can override the default
behavior to enable custom authentication for specific external user directories or for Native
Directory.
**Updating User Directory Configurations**

You must update the configuration of the user directory for which custom authentication must be enabled.

➢ To update user directory configuration:

1. Start Foundation Services.
2. Log in to Shared Services Console as a Shared Services administrator.
3. Select Administration, and then Configure User Directories.
4. In the Defined User Directories screen, select the user directory for which you want to change the custom authentication setting.

   **Note:** EPM System uses only the user directories included in the search order.

5. Click Edit.
6. Select Show Advanced Options.
7. In Custom Module, select Authentication Module to enable custom module for the current user directory.
8. Click Finish.
9. Repeat this procedure to update the configuration of other user directories in the search order.

**Updating Security Options**

Ensure that CustomAuth.jar is available in EPM_ORACLE_HOME/user_projects/domains/WEBLOGIC_DOMAIN/lib, typically, Oracle/Middleware/user_projects/domains/EPMSYSTEM/lib before starting the following procedure.

➢ To update security options:

1. Log in to Shared Services Console as a Shared Services administrator.
2. Select Administration, and then Configure User Directories.
4. Select Show Advanced Options.
5. In Authentication Module, enter the fully qualified class name of the custom authentication module that should be used to authenticate users on all user directories for which the custom authentication module is selected. For example, com.mycompany.epm.CustomAuthenticationImpl.
6. Click OK.

**Testing Your Deployment**

If Native Directory is not configured for custom authentication, do not use Native Directory users to test custom authentication.
Note: It is your responsibility to identify and correct any issues with the custom authentication module. Oracle assumes that your custom module works flawlessly to map a user from the user directory used by the custom module to a user on a custom authentication-enabled user directory available in EPM System search order.

To test your deployment, log in to EPM System using user credentials from the user directory, for example, an RSA SecurID infrastructure, used by the custom module. These credentials may be different from the EPM System credentials.

Your implementation is considered successful if EPM System products allow you to access their resources. An error indicating that the user was not found is not always an indicator of an unsuccessful implementation. In such cases, verify that the credentials that you entered are present in the custom user store and that a matching user is present in one of the custom authentication-enabled user directories in EPM System search order.

To test custom authentication:

1. Ensure that EPM System products are running.
2. Access an EPM System product; for example, EPM Workspace.
3. Log in as a user defined on a user directory for which custom authentication is enabled.
   a. In Username, enter your user identifier; for example, an RSA User ID.
   b. In Password, enter a password; for example; an RSA PIN.
   c. Click Login.
4. Verify that you can access EPM System product resources.
Using a Custom Authentication Module
In This Chapter

Implementing SSL .......................................................... 89
Changing the Admin Password ........................................... 89
Regenerating Encryption Keys .......................................... 90
Changing Database Passwords ......................................... 90
Securing Cookies ................................................................ 91
Reducing SSO Token Timeout ........................................... 91
Reviewing Security Reports .................................................. 91
Customizing Authentication System for Strong Authentication .......................... 92
Turning off Detailed Financial Management Error Messages ......................... 92
Encrypting UDL File (Financial Management) ................................... 92
Disabling EPM Workspace Debugging Utilities ................................. 93
Changing Default Web Server Error Pages .................................. 93
Support for Third-Party Software ......................................... 93

Implementing SSL

SSL uses a cryptographic system that encrypts data. SSL creates a secure connection between a client and a server, over which data can be sent securely.

To secure your EPM System environment, secure all communication channels used by your web applications and user directory connections using SSL. See Chapter 2, “SSL-Enabling EPM System Components”.

Changing the Admin Password

The default Native Directory admin user account provides access to all Shared Services functions. This password is set when you deploy Foundation Services. You must periodically change the password of this account.

Edit the admin user account to change the password. See “Modifying User Accounts” in the Oracle Enterprise Performance Management System User and Role Security Guide.
Regenerating Encryption Keys

Use the Shared Services Console to periodically regenerate the following:

- Single Sign-On Token

  **Caution!** Taskflows used by Financial Management; Oracle Hyperion EPM Architect and Oracle Hyperion Profitability and Cost Management are invalidated when you generate a new keystore. After regenerating the keystore, open and save the taskflows to revalidate them.

- Trusted Services key
- Provider Configuration key


Changing Database Passwords

Periodically change the password for all EPM System product databases. The procedure for changing the database password in Shared Services Registry is detailed in this section.

For detailed procedures to change an EPM System product database password, see the *Oracle Enterprise Performance Management System Installation and Configuration Guide*.

To change EPM System product database passwords in Shared Services Registry:

1. Using the database administration console, change the password of the user whose account was used to configure EPM System product database.
2. Stop EPM System products (web applications, services and processes).
3. Using the EPM System Configurator, reconfigure the database using one of the following procedures.

   **Shared Services Only:**
   
   a. From the Foundation tasks in EPM System Configurator, select **Configure Database**.
   b. On the Shared Services and Registry Database Configuration page, select **Connect to a previously configured Shared Services database**.
   c. Specify the new password of the user whose account was used to configure Shared Services database. Do not change any other settings.
   d. Continue the configuration and click **Finish** when you are done.

   **EPM System Products Other Than Shared Services:**
Follow these steps for the EPM System products deployed on the current server only.

a. From the configuration task list of the product in EPM System Configurator, select **Configure Database**.
b. On the Database Configuration page, select **Perform 1st-time configuration of database**.
c. Specify the new password of the user whose account was used to configure EPM System product database. Do not change any other settings.
d. Click **Next**.
e. Select **Reuse the existing database**.
f. Continue the configuration, and click **Finish** when you are done.

See the Oracle Enterprise Performance Management System Installation and Configuration Guide for detailed instructions.

4 Start EPM System products and services.

**Securing Cookies**

EPM System web application sets a cookie to track the session. While setting a cookie, especially a session cookie, the server can set the secure flag, which forces the browser to send the cookie over a secure channel. This behavior reduces the risk of session hijacking.

Note: Secure cookies only if EPM System products are deployed in an SSL-enabled environment.

Modify the WebLogic Server session descriptor to secure WebLogic Server cookies. Set the value of cookieSecure attribute in the session-param element to true.

**Reducing SSO Token Timeout**

Default SSO token timeout is 480 minutes. You should reduce the SSO token timeout, for example, to 60 minutes to minimize token reuse if it is exposed. See “Setting Security Options” in the Oracle Enterprise Performance Management System User and Role Security Guide.

**Reviewing Security Reports**

The Security Report contains audit information related to the security tasks for which auditing is configured. Generate and review this report from Shared Services Console on a regular basis, especially to identify failed login attempts across EPM System products and provisioning changes. Select **Detailed View** as a report generation option to group the report data based on attributes that were modified and the new attribute values. See “Generating Reports” in the Oracle Enterprise Performance Management System User and Role Security Guide.
Customizing Authentication System for Strong Authentication

You can use a custom authentication module to add strong authentication to EPM System. For example, you can use RSA SecurID two-factor authentication in nonchallenge response mode. The custom authentication module is transparent for thin and thick clients and does not require client-side deployment changes. See Chapter 4, “Using a Custom Authentication Module”.

Turning off Detailed Financial Management Error Messages

You can hide detailed Financial Management error messages containing technical information from users by updating Windows registry entries.

1. To hide error messages containing detailed technical information:
   1. On Windows server that hosts Financial Management, launch the Windows Registry Editor.
   2. Navigate to `HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\Hyperion Financial Management`
   3. Create a new DWORD value using these settings:
      - Value name: DisableTechnicalErrorMessage
      - Value data: 1 (set this to 0 to display detailed messages)

   4. On the Windows server that hosts the IIS Server that hosts Financial Management, launch the Windows Registry Editor.

   5. Navigate to `HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\Hyperion Financial Management\web`

   6. Create a new DWORD value using these settings:
      - Value name: DisableAspTechnicalErrorMessage
      - Value data: 1 (set this to 0 to display detailed messages)

Encrypting UDL File (Financial Management)

While configuring Financial Management, EPM System Configurator creates an unencrypted UDL file by default. You can encrypt this file by selecting an option in the Advanced Database Options page of the Oracle Hyperion Enterprise Performance Management System Configurator or by running the `EncryptHFMUDL` utility after configuration is complete.

See “Encrypting UDL Files” in Oracle Enterprise Performance Management System Installation and Configuration Guide.
Disabling EPM Workspace Debugging Utilities

- For troubleshooting purposes, EPM Workspace ships with uncrunched JavaScript files. For security purposes, you should remove these uncrunched JavaScript files from your production environment:
  - Create a backup copy of `EPM_ORACLE_HOME/common/epmstatic/wspace/js` directory.
  - Except for the file `DIRECTORY_NAME.js`, delete the `.js` files from each subdirectory of `EPM_ORACLE_HOME/common/epmstatic/wspace/js`.
    Each subdirectory contains a `.js` file that bears the name of the directory. For example, `EPM_ORACLE_HOME/common/epmstatic/wspace/js/com/hyperion/bpm/web/common` contains `Common.js`. Remove all `.js` files except the one that bears the name of the directory, in this case; `Common.js`.

- EPM Workspace provides some debug utilities and test applications, which become accessible if EPM Workspace is deployed in debug mode. For security purposes, administrators should turn off client side debugging in EPM Workspace.

  To turn off debugging mode:
  1. Log in to Oracle Hyperion Enterprise Performance Management Workspace, as administrator.
  2. Select Navigate, then Administer, and then Workspace Server Settings.
  4. Click OK.

Changing Default Web Server Error Pages

When application servers are not available to accept requests, the web server plug-in for the back-end application server (for example, Oracle HTTP Server plug-in for Oracle WebLogic Server) returns a default error page that displays plug-in build information. Web servers display their default error page on other occasions as well. Attackers can use this information to find known vulnerabilities from public web sites.

Customize the error pages (of web application server plug-in and web server) so that they do not contain information about production system components, for example, server version, server type, plug-in build date, and plug-in type. Consult your application server and web server vendor documentation for more information.

Support for Third-Party Software

Oracle acknowledges and supports the backward-compatibility assertions made by third-party vendors. Therefore, where vendors assert backward-compatibility, subsequent maintenance releases and service packs may be used. If an incompatibility is identified, Oracle will specify a
patch release on which the product should be deployed (and remove the incompatible version from the supported matrix) or provide a maintenance release or service fix to the Oracle product.

**Server-side Updates:** Support for upgrades to third-party server-side components is governed by the Subsequent Maintenance Release Policy. Typically, Oracle supports upgrading third-party server-side components to the next maintenance release of service pack of the currently supported release. Upgrades for the next major release are not supported.

**Client-side updates:** Oracle supports automatic updates to client components, including updates to the next major release of third-party client components. For example, you can update the browser JRE version from 1.5 to 1.6.
Sample Code 1

Note: Your custom authentication code should not use log4j for error logging. If the custom authentication code that you used in a previous release used log4j, you must remove it from the code before using it with this release.

The following code snippet is an empty implementation of the custom module:

```java
package com.hyperion.css.custom;

import java.util.Map;
import com.hyperion.css.CSSCustomAuthenticationIF;

public class CustomAuthenticationImpl implements CSSCustomAuthenticationIF {
    public String authenticate(Map context, String userName, String password) throws Exception {
        try {
            // Custom code to find and authenticate the user goes here.
            // The code should do the following:
            // if authentication succeeds:
            //   set authenticationSuccessFlag = true
            //   return authenticatedUserName
            // if authentication fails:
            //   log an authentication failure
            //   throw authentication exception
        } catch (Exception e) {
            // Custom code to handle authentication exception goes here
            // Create a new exception, set the root cause
            // Set any custom error message
            // Return the exception to the caller
        }
        return authenticatedUserName;
    }

    public String authenticate(String userName, String password) throws Exception {
        return authenticate(null, userName, password);
    }
}
```
Input parameters:

- **Context**: A map that contains key-value pair of locale information
- **User name**: An identifier that uniquely identifies the user to the user directory where the custom module authenticates the user. The user enters the value of this parameter while logging into an EPM System product.
- **Password**: The password set for the user in the user directory where the custom module authenticates the user. The user enters the value of this parameter while logging into an EPM System product.

### Sample Code 2

The following sample code demonstrates custom authentication of users using user name and password contained in a flat file. You must initialize user and password lists in the class constructor to make custom authentication work.

```java
package com.hyperion.css.security;

import java.util.Map;
import java.util.HashMap;
import com.hyperion.css.CSSCustomAuthenticationIF;
import java.io.*;

public class CSSCustomAuthenticationImpl implements CSSCustomAuthenticationIF{
    static final String DATA_FILE = "datafile.txt";

    /**
     * authenticate method includes the core implementation of the
     * Custom Authentication Mechanism. If custom authentication is
     * enabled for the provider, authentication operations
     * are delegated to this method. Upon successful authentication,
     * this method returns a valid user name, using which EPM System
     * retrieves the user from a custom authentication enabled provider.
     * User name can be returned in the format username@providerName,
     * where providerName indicates the name of the underlying provider
     * where the user is available. authenticate method can use other
     * private methods to access various core components of the
     * custom authentication module.
     *
     * @param context
     * @param userName
     * @param password
     * @return
     * @throws Exception
     */
    Map users = null;

    public CSSCustomAuthenticationImpl(){
        users = new HashMap();
    }
}
```
InputStream is = null;
BufferedReader br = null;
String line;
String[] userDetails = null;
String userKey = null;
try{
    is = CSSCustomAuthenticationImpl.class.getResourceAsStream(DATA_FILE);
    br = new BufferedReader(new InputStreamReader(is));
    while(null != (line = br.readLine())){
        userDetails = line.split(":");
        if(userDetails != null && userDetails.length==3){
            userKey = userDetails[0] + " : " + userDetails[1];
            users.put(userKey, userDetails[2]);
        }
    }
} catch(Exception e){
    // log a message
}
finally{
    try{
        if(br != null) br.close();
        if(is != null) is.close();
    }
    catch(IOException ioe){
        ioe.printStackTrace();
    }
}
/* Use this authenticate method snippet to return username from a flat file */

public String authenticate(Map context, String userName, String password) throws Exception{
    //userName : user input for the userName
    //password : user input for password
    //context : Map, can be used to additional information required by
            // the custom authentication module.

    String authenticatedUserKey = userName + " : " + password;

    if(users.get(authenticatedUserKey)!=null)
        return(String)users.get(authenticatedUserKey);
    else throw new Exception("Invalid User Credentials");
}
/* Refer to this authenticate method snippet to return username in
username@providername format */

public String authenticate(Map context, String userName, String password) throws Exception{
    //userName : user input for userName
    //password : user input for password
    //context : Map can be used to additional information required by
            // the custom authentication module.
// Your code should uniquely identify the user in a custom provider and in a configured user directory in Shared Services. EPM Security expects you to append the provider name to the user name. Provider name must be identical to the name of a custom authentication-enabled user directory specified in Shared Services.

// If invalid arguments, return null or throw exception with appropriate message
// set authenticationSuccessFlag = false

String authenticatedUserKey = userName + ":*" + password;
if(users.get(authenticatedUserKey) != null) {
    String userNameStr = (new StringBuffer())
        .append((String)users.get(authenticatedUserKey))
        .append("@").append(PROVIDER_NAME).toString();
    return userNameStr;
} else throw new Exception("Invalid User Credentials");
}

---

**Data File for Sample Code 2**

Ensure that the data file is named `datafile.txt`, which is the name used in the sample code, and that it is included in the Java archive that you create.

Use the following as the contents of the flat file that is used as the custom user directory to support the custom authentication module implemented by Sample Code 2 (see “Sample Code 2” on page 96):

```
xyz:password:admin
test1:password:test1@LDAP1
test1:password:test1
test1@LDAP1:password:test1@LDAP1
test1@1:password:test1
user1:Password2:user1@SunONE1
user1_1:Password2:user1
user3:Password3:user3
DS_User1:Password123:DS_User1@MSAD1
DS_User1:Password123:DS_User1
DS_User1@1:Password123:DS_User1
```

Use the following as the contents of the flat file that is used as the custom user directory if you plan to return user name in `username@providername` format:

```
xyz:password:admin
test1:password:test1
test1@1:password:test1
user1_1:Password2:user1
user3:Password3:user3
DS1_1G100U_User61_1:Password123:DS1_1G100U_User61
DS1_1G100U_User61_1@1:Password123:DS1_1G100U_User61
TUser:password:TUser
```
Implementing a Custom Login Class

In This Appendix

Custom Login Class Sample Code ................................................................. 99
Deploying a Custom Login Class ................................................................. 102

EPM System provides
com.hyperion.css.sso.agent.X509CertificateSecurityAgentImpl to extract the
user identity (DN) from x509 certificates.

If you must derive user identity from an attribute in the certificate other than DN, you must
develop and implement a custom login class similar to
com.hyperion.css.sso.agent.X509CertificateSecurityAgentImpl, as described in
this appendix.

Custom Login Class Sample Code

This sample code illustrates the implementation of the default
should customize the parseCertificate(String sCertificate) method of this
implementation to derive the user name from a certificate attribute other than DN:

```java
package com.hyperion.css.sso.agent;

import java.io.ByteArrayInputStream;
import java.io.UnsupportedEncodingException;
import java.security.Principal;
import java.security.cert.CertificateException;
import java.security.cert.CertificateFactory;
import java.security.cert.X509Certificate;
import com.hyperion.css.CSSSecurityAgentIF;
import com.hyperion.css.common.configuration.*;
import java.util.HashMap;
import java.util.Locale;
import java.util.Map;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * X509CertificateAuthImpl implements the CSSSecurityAgentIF interface It accepts
 * the X509 certificate of the authenticated user from the Web Server via a
 */
```
* header, parses the certificate, extracts the DN of the User and
* authenticates the user.
*/
public class X509CertificateSecurityAgentImpl implements CSSSecurityAgentIF
{
    static final String IDENTITY_ATTR = "CN";
    String g_userDN = null;
    String g_userName = null;
    String hostAdrress = null;
    /**
     * Returns the User name (login name) of the authenticated user,
     * for example demouser. See CSS API documentation for more information
     */
    public String getUserName(HttpServletRequest req, HttpServletResponse res)
        throws Exception
    {
        hostAdrress = req.getServerName();
        String certStr = getCertificate(req);
        String sCert = prepareCertificate(certStr);
        /* Authenticate with a CN */
        parseCertificate(sCert);
        /* Authenticate if the Login Attribute is a DN */
        if (g_userName == null)
        {
            throw new Exception("User name not found");
        }
        return g_userName;
    }

    /**
     * Passing null since this is a trusted Security agent authentication
     * See Security API documentation for more information on CSSSecurityAgentIF
     */
    public String getPassword(HttpServletRequest req, HttpServletResponse res)
        throws Exception
    {
        return null;
    }

    /**
     * Get the Certificate sent by the Web Server in the HYPLOGIN header.
     * If you pass a different header name from the Web server, change the
     * name in the method.
     */
    private String getCertificate(HttpServletRequest request)
    {
        String cStr = (String)request
            .getHeader(CSSConfigurationDefaults.HTTP_HEADER_HYPLOGIN);
        return cStr;
    }

    /**
     * The certificate sent by the Web server is a String.
     * Put a "\n" in place of whitespace so that the X509Certificate
     */
private String prepareCertificate(String gString) {
    String str1 = null;
    String str2 = null;

    str1 = gString.replace("-----BEGIN CERTIFICATE-----", "");
    str2 = str1.replace("-----END CERTIFICATE-----", "");
    String certStrWithNL = "-----BEGIN CERTIFICATE-----" + str2.replace(" ", "\n") + "-----END CERTIFICATE-----";
    return certStrWithNL;
}

/**
 * Parse the certificate
 * 1. Create X509Certificate using the certificateFactory
 * 2. Get the Principal object from the certificate
 * 3. Set the g_userDN to a certificate attribute value (DN in this sample)
 * 4. Parse the attribute (DN in this sample) to get a unique username
 */
private void parseCertificate(String sCertificate) throws Exception {
    X509Certificate cert = null;
    String userID = null;
    try {
        X509Certificate clientCert = (X509Certificate)CertificateFactory.getInstance("X.509")
            .generateCertificate(new ByteArrayInputStream(sCertificate.getBytes("UTF-8")));

        if (clientCert != null) {
            Principal princDN = clientCert.getSubjectDN();
            String dnStr = princDN.getName();
            g_userDN = dnStr;
            int idx = dnStr.indexOf(",");
            userID = dnStr.substring(3, idx);
            g_userName = userID;
        }
    }
    catch (CertificateException ce) {
        throw ce;
    }
    catch (UnsupportedEncodingException uee) {
        throw uee;
    }
} //end of getUserNameFromCert
} // end of class
Deploying a Custom Login Class

To implement the custom login class, complete the following steps:

- Create and test the custom login class. Ensure that you do not have any references to log4j in your code. See “Custom Login Class Sample Code” on page 99.
  
  You can use any name for your custom class.

- Package the custom login class into CustomAuth.jar

- Copy CustomAuth.jar into EPM_ORACLE_HOME/user_projects/domains/WEBLOGIC_DOMAIN/lib, typically, Oracle/Middleware/user_projects/domains/EPMSystem/lib.

  **Note:** If you are upgrading from Release 11.1.2.0 or 11.1.2.1 that had an implementation of custom login class, move CustomAuth.jar from EPM_ORACLE_HOME/common/jlib/11.1.2.0 into EPM_ORACLE_HOME/user_projects/domains/WEBLOGIC_DOMAIN/lib.

Oracle recommends that you enable Client Certificate Authentication if you are using a custom login class.
About the Update Native Directory Utility

Native Directory contains information that references user and group identities defined in many external user directories. For example, Native Directory groups can contain users defined in external user directories. Changes in external user directories, such as the deletion of a user account or the migration of users from an external user directory to another (see Appendix D, “Migrating Users and Groups Across User Directories”) may cause stale data within Native Directory because EPM System security is not synchronized to be aware of such changes. In such cases, use Update Native Directory Utility to identify and remove stale data from Native Directory.

Update Native Directory Utility Installation Location

The Update Native Directory Utility is installed in EPM_ORACLE_HOME/common/utilities/UpdateNativeDir; for example, in C:\Oracle\Middleware\EPMSystem11R1\common\utilities\UpdateNativeDir on a Windows server.

Update Native Directory Utility Options


Caution! Update Native Directory Utility considers provisioning data of users and groups from user directories that are not included in the search order as stale data. If you need to retain such data, you must remove it from CSS_MIGRATION_DELETE_LIST.csv.


### Using Update Native Directory Utility

Generally, you complete the following steps to manage stale Native Directory data:

- “Identifying Stale Data” on page 104
- “Deleting Stale Data” on page 105
- Resolve ambiguous identities. Ambiguous identities are identities that the utility failed to resolve. You must manually resolve these identities.

### Updating the Update Native Directory Utility Settings

Modify parameter values in `updateNativeDir.bat` (Windows) or `updateNativeDir.sh` (UNIX) located in `EPM_ORACLE_HOME/common/utilities/UpdateNativeDir`.

1. **To update utility settings:**
   - **Using a text editor**, open `updateNativeDir.bat` (Windows) or `updateNativeDir.sh` (UNIX) located in `EPM_ORACLE_HOME/common/utilities/UpdateNativeDir`.
   - **Update the value of EPM_ORACLE_INSTANCE** to reflect the instance location in your environment. By default, `EPM_ORACLE_INSTANCE` is `C:\Oracle\Middleware\user_projects\epmsystem1` (Windows).
   - **Save and close the file.**

### Identifying Stale Data

Run Update Native Directory Utility without specifying options to generate `CSS_MIGRATION_DELETE_LIST.csv` that identifies the stale identities in Native Directory that can be deleted.
To identify stale Native Directory data:


2. Using a command prompt window or console on the server that hosts an EPM System product, navigate to `EPM_ORACLE_HOME/common/utilities/UpdateNativeDir`; for example, to `C:\Oracle\Middleware\EPMS11R1\common\utilities\UpdateNativeDir` on a Windows server.

3. Execute a command:
   
   ```
   - updateNativeDir -cssLocation LOCATION_OF_CSS.xml (Windows)
   - updateNativeDir.sh -cssLocation LOCATION_OF_CSS.xml (UNIX)
   ```

   In the preceding command, `LOCATION_OF_CSS.xml` denotes the absolute path to a `CSS.xml` that you generated from the Oracle Hyperion Shared Services Registry; for example, `C:\CSS.xml` on a Windows server.

4. Enter 1 at the following Update Native Directory Utility query:
   
   ```
   Do you want to proceed? [0->No/1->Yes] :
   ```

Deleting Stale Data

Before deleting stale data, verify the contents of `CSS_MIGRATION_DELETE_LIST.csv`. See “Identifying Stale Data” on page 104.

**Note:** Update Native Directory Utility does not delete stale Native Directory identities that are referenced from an external user directory to which it cannot establish a connection.

**Caution!** The delete operation removes provisioning data that references users and groups from external user directories that are not included in Shared Services search order.

To delete stale Native Directory data:


2. Using a command prompt window or console on the server that hosts an EPM System product, navigate to `EPM_ORACLE_HOME/common/utilities/UpdateNativeDir`; for example, to `C:\Oracle\Middleware\EPMS11R1\common\utilities\UpdateNativeDir` on a Windows server.

3. Execute a command. For a list of options you can specify, see “Update Native Directory Utility Options” on page 103.

   **Note:** You can combine the `-noprompt` and `-cssLocation` directives with these commands.
- updateNativeDir -delete PATH_OF_DELETE_LIST
- updateNativeDir -delete all

In this command, `PATH_OF_DELETE_LIST` refers to the absolute location of CSS_MIGRATION_DELETE_LIST.csv; for example, C:\Oracle\Middleware\EPMSystem11R1\common\utilities\UpdateNativeDir\logs\security-migration\CSS_MIGRATION_DELETE_LIST.csv on a Windows server.

4 Enter 1 at the following Update Native Directory Utility query:
Do you want to proceed? 0->No/1->Yes :

**Log Files Generated by Update Native Directory Utility**

By default, Update Native Directory Utility creates log files in `EPM_ORACLE_HOME/common/utilities/UpdateNativeDir/logs/security-migration`.

- CSSMigration-Ambiguous_time_stamp.log lists ambiguous identities that Update Native Directory Utility could not resolve. You must manually update the identities listed in this file.
- CSSMigration-Deleted_time_stamp.log lists the identities that Update Native Directory Utility deleted from Native Directory.
- CSSMigration-Updated_time_stamp.log lists the Native Directory identities that Update Native Directory Utility updated in Native Directory to reflect the changes to the identity in an external user directory.
- CSSMigration-Ignored_time_stamp.log lists the entries on which no action was taken because they needed no update.
Migrating Users and Groups Across User Directories

In This Appendix

Overview .................................................................................................. 107
Prerequisites .......................................................................................... 107
Migration Procedure .................................................................................. 108
Product-Specific Updates ........................................................................... 111

Overview

There are many scenarios that may cause the user and group identities of provisioned EPM System users to become stale. EPM System components become inaccessible if the provisioning information available to them is stale. Scenarios that may create stale provisioning data include:

- Retiring a user directory: Organizations may retire a user directory after moving users to another.
- Version upgrade: User directory version upgrade may involve changes in host machine name or operating system environments requiring.
- Vendor change: Organizations may discontinue the use of a user directory in favor of a user directory from another vendor. For example, an organization may replace its Oracle Internet Directory with a SunONE Directory Server.

Note: In this appendix, the user directory that you are phasing out is referred to as the source user directory, and the user directory to which you moved the user accounts is referred to as the target user directory.

Prerequisites

- EPM System users and groups whose provisioning data is being migrated across user directories must be available in the target user directory.
  
  Group relationships that exist in the source user directory must be maintained in the target user directory.
- User names of EPM System users must be identical across source and target user directories.
Migration Procedure

Subtopics
- Export Native Directory Data
- Prepare EPM System for Migration
- Restart EPM System
- Edit Import Files
- Import Updated Data
- Run Update Native Directory Utility

Export Native Directory Data

Use Oracle Hyperion Enterprise Performance Management System Lifecycle Management to export the following from Native Directory:
- Native Directory Groups
- Assigned roles
- Delegated lists

Lifecycle Management creates multiple export files, generally in `EPM_ORACLE_INSTANCE/import_export/USER_NAME/EXPORT_DIR/resource/Native Directory`, where `USER_NAME` is the identity of the user; for example, `admin@Native Directory`, who performed the export operation and `EXPORT_DIR` is the name of the export directory. Typically, these files are created:
- Groups.csv
- Roles.csv
- Delegated Lists.csv
- `Assigned Roles/PROD_NAME.csv` for each deployed application, where `PROD_NAME` is the name of an EPM System component; for example, `Shared Services`.

Note: See the Oracle Enterprise Performance Management System Lifecycle Management Guide for detailed instructions on exporting data using Lifecycle Management.

To export provisioning data from Native Directory:

1. In the View pane of Shared Services Console, select Shared Services application within Foundation application group.
2. Select the type of artifacts for which you want to export provisioning information.
3. Select Define Migration.
4. Set source options and then click Next.
5. Enter a file system location for storing export files and then click Next.
6. Click Next in Destination Options.
Click Execute Migration.

Prepare EPM System for Migration

Subtopics

- Add the Target User Directory as an External User Directory
- Change the Search Order of the Target User Directory

Add the Target User Directory as an External User Directory

Add the target user directory as an external user directory in EPM System if you moved the user accounts from the source user directory to a different user directory. For example, if you moved the user accounts from Oracle Internet Directory to SunONE Directory Server, add SunONE Directory Server as an external user directory. See “Chapter 3, Configuring User Directories” in the Oracle Enterprise Performance Management System User and Role Security Guide.

Note: Ensure that the target user directory contains user accounts and groups for all EPM System users whose data is being migrated from the source user directory.

If you moved the users to a user directory that is already defined as an external user directory, verify that the user accounts are visible to Shared Services. You can do this by searching for users from Shared Services Console. See “Searching for Users, Groups, Roles, and Delegated Lists” in the Oracle Enterprise Performance Management System User and Role Security Guide.

While configuring the target user directory as an external user directory, verify that the Login Attribute property points to the attribute whose value was originally used as the user name in the source user directory. See “Prerequisites” on page 107.

Change the Search Order of the Target User Directory

Note: If the target user directory name is identical to the source directory name, you must delete the source user directory from EPM System configuration.

Shared Services assigns a lower search order priority to a newly added user directory as compared to the search order assigned to existing directories. Change the search order so that the target user directory has a higher search order priority than the source user directory. This allows Oracle Hyperion Shared Services to discover users in the target user directory before searching the source. See “Managing the User Directory Search Order” in the Oracle Enterprise Performance Management System User and Role Security Guide.
Restart EPM System

Restart Oracle Hyperion Foundation Services and other EPM System components to enforce the changes you made.

Edit Import Files

**Note:** This step is not required if the target user directory name in EPM System configuration is identical to the source user directory name.

You use the export files that Lifecycle Management created as the source for recreating the data in Native Directory. The export files are generated in the directory that you specified while exporting data from Native Directory. See “Export Native Directory Data” on page 108.

In each export file, replace all references to the source user directory with references to the target user directory. Generally, you edit the assigned roles export files and, optionally, the following files.

- **Groups.csv** if users from the source user directory are members of Native Directory groups.
- **Delegated Lists.csv** if users from the source user directory are assigned to delegated lists.

The import files are in `{EPM_ORACLE_INSTANCE}/import_export/{USER_NAME}/EXPORT_DIR/resource/Native Directory`, where `{USER_NAME}` is the identity of the user; for example, admin@Native Directory, who performed the export operation, and `EXPORT_DIR` is the name of the export directory.

To edit an import file:

1. Using a text editor, open an import file.
2. Replace the name of the source user directory with the name of the target user directory as displayed in the Directory Name column in the Defined User Directories screen.
3. Save and close the import file.

Import Updated Data

Run Lifecycle Management with `create/update` option to import the data you exported earlier from Native Directory. See “Export Native Directory Data” on page 108.

**Note:** See the *Oracle Enterprise Performance Management System Lifecycle Management Guide* for detailed instructions on importing data using Oracle Hyperion Enterprise Performance Management System Lifecycle Management.
To import updated provisioning data into Native Directory:

1. In the View pane of Oracle Hyperion Shared Services Console, expand File System.
2. Select the file system location of the import files.
3. Select the type of artifacts for which you want to import provisioning information.
4. Click Define Migration.
5. In Source Options, click Next.
6. In Destination, click Next.
7. In Destination Options, verify that Import Operation Type is set to create/update.
8. Click Next.
9. Click Execute Migration.

Run Update Native Directory Utility

Clean stale data from Native Directory by running the Update Native Directory Utility. See Appendix C, “Using the Update Native Directory Utility.”

Product-Specific Updates

Caution! Oracle recommends that you back up the user and group data in the repository used by the Oracle Enterprise Performance Management System component before starting product-specific updates. After updating information in the local product repository, you can revert to the old user and group data in the local product repository from backups only.

Planning

Planning stores information about provisioned users and groups in the Planning repository. If a user identity was changed in Native Directory as a result of migrating users and groups across user directories, you must synchronize the information in the Planning repository with that in Native Directory by selecting Migrate Users/Groups. This button is available in Oracle Hyperion Planning when assigning access to data forms, members, or task lists.

Financial Management

Financial Management records information about users and groups provisioned to access objects in a local Financial Management repository. If user and group information in Native Directory has changed as a result of migrating users and groups across user directories, you must synchronize the information in the Oracle Hyperion Financial Management repository with that in Native Directory.
Reporting and Analysis

Reporting and Analysis uses the syncCSSId utility to synchronize user and group identities stored in its relational database to reflect the identities available in Native Directory. You must run this utility before users are allowed to access Oracle Hyperion Reporting and Analysis after migrating provisioning data in Native Directory. The syncCSSId utility is installed in $EPM_ORACLE_INSTANCE/bin/ReportingAnalysis/syncCSSId directory; for example, C:/Oracle/Middleware/user_projects/epmsystem1/bin/ReportingAnalysis/syncCSSId.

See $EPM_ORACLE_INSTANCE/bin/ReportingAnalysis/syncCSSId/ReadmeSyncCSSId_BI.txt for detailed instructions to run the syncCSSId utility.
access permissions  A set of operations that a user can perform on a resource.

aggregated role  A custom role that aggregates multiple predefined roles within a Hyperion product.

application  1) A software program designed to run a specific task or group of tasks such as a spreadsheet program or database management system; 2) A related set of dimensions and dimension members that are used to meet a specific set of analytical requirements, reporting requirements, or both.

Application Migration Utility  A command-line utility for migrating applications and artifacts.

artifact  An individual application or repository item; for example, scripts, forms, rules files, Interactive Reporting documents, and financial reports. Also known as an object.

authentication  Verification of identity as a security measure. Authentication is typically based on a user name and password. Passwords and digital signatures are forms of authentication.

automated stage  A stage that does not require human intervention; for example, a data load.

backup  A duplicate copy of an application instance.

business process  A set of activities that collectively accomplish a business objective.

context variable  A variable that is defined for a particular task flow to identify the context of the taskflow instance.

external authentication  Logging on to Oracle EPM System products with user information stored outside the application. The user account is maintained by the EPM System, but password administration and user authentication are performed by an external service, using a corporate directory such as Oracle Internet Directory (OID) or Microsoft Active Directory (MSAD).

filter  A constraint on data sets that restricts values to specific criteria; for example, to exclude certain tables, metadata, or values, or to control access.

group  A container for assigning similar access permissions to multiple users.

identity  A unique identification for a user or group in external authentication.

integration  A process that is run to move data between Oracle’s Hyperion applications using Shared Services. Data integration definitions specify the data moving between a source application and a destination application, and they enable the data movements to be grouped, ordered, and scheduled.

lifecycle management  The process of migrating an application, a repository, or individual artifacts across product environments.

link  1) A reference to a repository object. Links can reference folders, files, shortcuts, and other links; 2) In a taskflow, the point where the activity in one stage ends and another begins.

link condition  A logical expression evaluated by the taskflow engine to determine the sequence of launching taskflow stages.

load balancing  Distribution of requests across a group of servers, which helps to ensure optimal end user performance.

managed server  An application server process running in its own Java Virtual Machine (JVM).

manual stage  A stage that requires human intervention.

migration  The process of copying applications, artifacts, or users from one environment or computer to another; for example, from a testing environment to a production environment.
migration audit report  A report generated from the migration log that provides tracking information for an application migration.

migration definition file (.mdf)  A file that contains migration parameters for an application migration, enabling batch script processing.

migration log  A log file that captures all application migration actions and messages.

migration snapshot  A snapshot of an application migration that is captured in the migration log.

model  1) In data mining, a collection of an algorithm’s findings about examined data. A model can be applied against a wider data set to generate useful information about that data; 2) A file or content string containing an application-specific representation of data. Models are the basic data managed by Shared Services, of two major types: dimensional and nondimensional application objects; 3) In Business Modeling, a network of boxes connected to represent and calculate the operational and financial flow through the area being examined.

product  In Shared Services, an application type, such as Planning or Performance Scorecard.

project  An instance of Oracle’s Hyperion products grouped together in an implementation. For example, a Planning project may consist of a Planning application, an Essbase cube, and a Financial Reporting Server instance.

provisioning  The process of granting users and groups specific access permissions to resources.

repository  Storage location for metadata, formatting, and annotation information for views and queries.

role  The means by which access permissions are granted to users and groups for resources.

security agent  A Web access management provider (for example, Oracle Access Manager, Oracle Single Sign-On, or CA SiteMinder) that protects corporate Web resources.

security platform  A framework enabling Oracle EPM System products to use external authentication and single sign-on.

Shared Services Registry  The part of the Shared Services repository that manages EPM System deployment information for most EPM System products, including installation directories, database settings, computer names, ports, servers, URLs, and dependent service data.

single sign-on (SSO)  The ability to log on once and then access multiple applications without being prompted again for authentication.

stage  1) A task description that forms one logical step within a taskflow, usually performed by an individual. A stage can be manual or automated; 2) For Profitability, logical divisions within the model that represent the steps in the allocation process within your organization.

stage action  For automated stages, the invoked action that executes the stage.

sync  Synchronization of Shared Services and application models.

synchronized  The condition that exists when the latest version of a model resides in both the application and in Shared Services. See also model.

task list  A detailed status list of tasks for a particular user.

taskflow  The automation of a business process in which tasks are passed from one taskflow participant to another according to procedural rules.

taskflow definition  Business processes in the taskflow management system that consist of a network of stages and their relationships; criteria indicating the start and end of the taskflow; and information about individual stages, such as participants, associated applications, associated activities, and so on.

taskflow instance  A single instance of a taskflow including its state and associated data.

taskflow management system  A system that defines, creates, and manages the execution of a taskflow, including definitions, user or application interactions, and application executables.

taskflow participant  The resource that performs the task associated with the taskflow stage instance for both manual and automated stages.

token  An encrypted identification of one valid user or group on an external authentication system.
**transformation** 1) A process that transforms artifacts so that they function properly in the destination environment after application migration; 2) In data mining, the modification of data (bidirectionally) flowing between the cells in the cube and the algorithm.

**upgrade** The process of deploying a new software release and moving applications, data, and provisioning information from an earlier deployment to the new deployment.

**user directory** A centralized location for user and group information, also known as a repository or provider. Popular user directories include Oracle Internet Directory (OID), Microsoft Active Directory (MSAD), and Sun Java System Directory Server.