Oracle® Enterprise Performance Management System

Security Configuration Guide

Release 11.2
F13806-26
June 2024

Oracle Enterprise Performance Management System Security Configuration Guide, Release 11.2

F13806-26

Primary Author: EPM Information Development Team

Contents

Documentation Accessibility

Documentation Feedback

1 About EPM System Security

2 SSL-Enabling EPM System Components

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 that have purchased support 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

To provide feedback on this documentation, click the feedback button at the bottom of the page in any Oracle Help Center topic. You can also send email to epmdoc_ww@oracle.com.

1

About EPM System Security

Related Topics

- About EPM System
- Assumed Knowledge
- Security Infrastructure Components
- User Authentication
- Provisioning (Role-Based Authorization)
- Launching Shared Services Console

About EPM System

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

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 system administrators who configure, secure, and manage Oracle Enterprise Performance Management System components. 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
  - Access Management Systems, for example, Oracle Access Manager, and SiteMinder
  - Single sign-on (SSO) infrastructure; for example, Kerberos
- Knowledge of EPM System security concepts that are relevant to your organization

Security Infrastructure Components

Oracle Enterprise Performance Management System integrates several 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 components
- 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 in to SiteMinder and EPM System components.
- Use of Secure Socket Layer (SSL) to secure communication channels among EPM System components and clients

User Authentication

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

Authentication Components

The following sections describe the components that support SSO:

- Native Directory
- External User Directories

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)

Native Directory is accessed and managed using Oracle Hyperion Shared Services Console. See "Managing Native Directory" in the *Oracle Enterprise Performance Management System User Security Administration Guide*.

External User Directories

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

EPM System components 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.

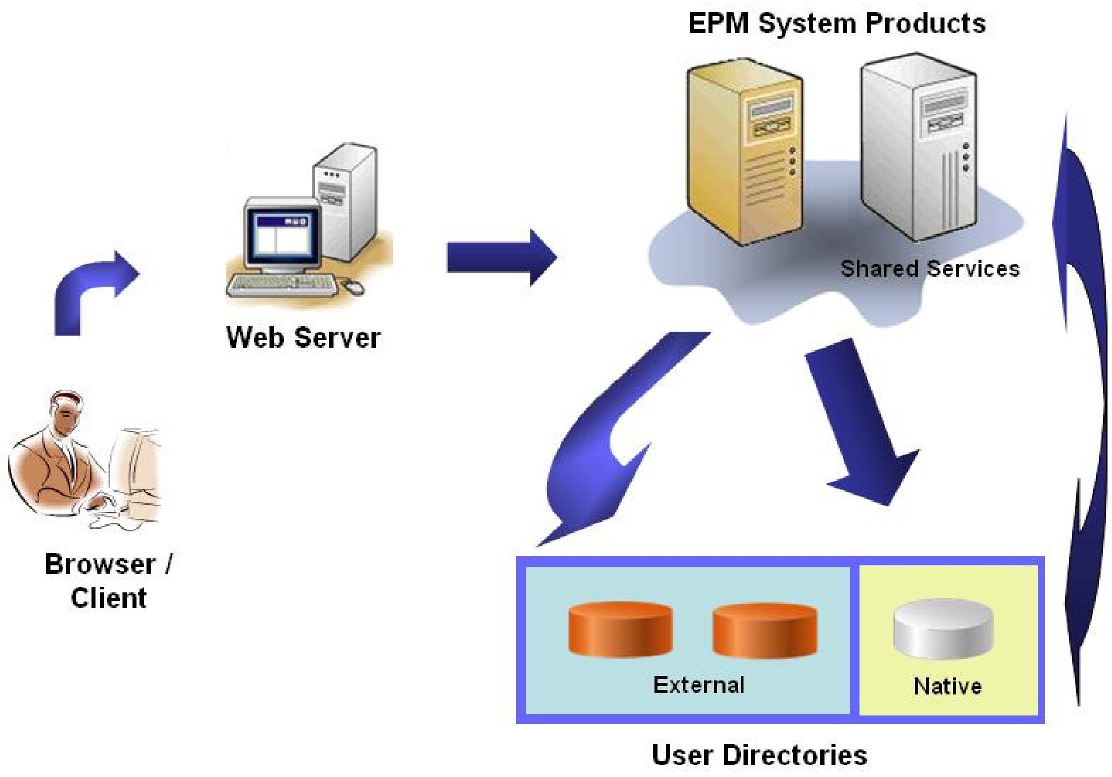
For a list of supported user directories, see the *Oracle Enterprise Performance Management System Certification Matrix* posted on the Oracle Fusion Middleware Supported System Configurations page on Oracle Technology Network (OTN).

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 a configured user directory. Generally, EPM System users are assigned to groups to facilitate provisioning.

Default EPM System Single Sign-on

EPM System supports 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 components.

The default SSO process is depicted in the following illustration.



1. Through a browser, users access an EPM System component login screen and enter a user name and password.

   The EPM System component queries the configured user directories (including Native Directory) to verify user credentials. Upon finding the matching user account in a user directory, the search is terminated, and the user's information is returned to the EPM System component.

   Access is denied if no user account is found in any configured user directory.

2. Using the retrieved user information, the EPM System component queries Native Directory to obtain provisioning details for the user.
3. EPM System component checks the Access Control List (ACL) in the component to determine the application artifacts that the user can access.

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

Single Sign-on from Access Management Systems

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

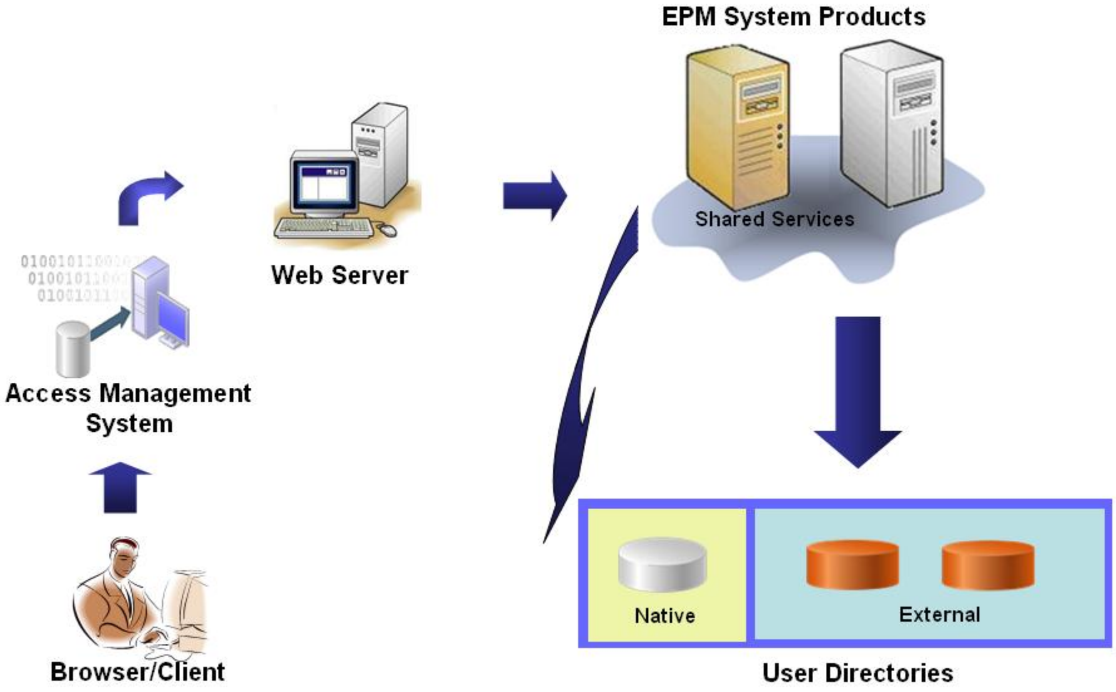
SSO from security agents is available for EPM System web applications only. In this scenario, EPM System components use the user information provided by the security agent to determine access permissions of users. To enhance security, Oracle recommends that direct access to servers be blocked by firewalls so 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. The access management system authenticates users and passes the login name to EPM System. EPM System verifies the login name against configured user directories.

See these topics.

- Single Sign-on from Oracle Access Manager
- OracleAS Single Sign-on
- SiteMinder SSO
- Kerberos Single Sign-on

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 components are defined as resources protected by the access management system.

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

   Information about the authenticated user is passed to the EPM System component, which accepts the information as valid.

   The access management system passes the user's login name (value of `Login Attribute`) to the EPM System component using an acceptable SSO mechanism. See Supported SSO Methods.

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

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

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

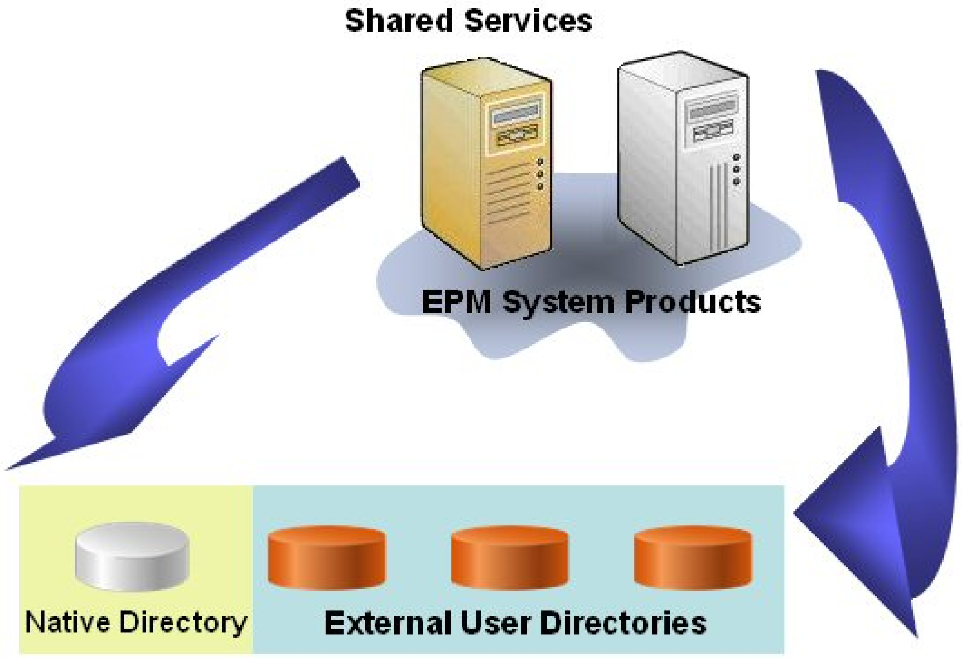
Provisioning (Role-Based Authorization)

Oracle Enterprise Performance Management 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 components enforce object-level ACLs to further refine user access to their artifacts, such as reports and members.

Each EPM System component provides several default roles tailored to various business needs. Each application belonging to an EPM System component inherits these roles. Predefined roles from the applications registered with Oracle Hyperion Shared Services are available from Oracle Hyperion 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 component queries user directories to determine the user's groups.
2. The EPM System component uses group and user information to retrieve the user's provisioning data from Shared Services. The component 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. A role 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 Oracle Hyperion Planning application. Aggregating roles can simplify the administration of applications that has 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.

Default EPM System Administrator

An administrator account, with default name `admin`, is created in Native Directory during the deployment process. This is the most powerful EPM System account and should be used only to set up a System Administrator, who is the Information Technology expert tasked with managing EPM System security and environment.

The user name and password of EPM System Administrator is set during Oracle Hyperion Foundation Services deployment. Because this account cannot be subjected to corporate account password policies, Oracle recommends that it be deactivated after creating a System Administrator account.

Generally, the default EPM System Administrator account is used to perform these tasks:

- Configure the corporate directory as an external user directory. See Configuring User Directories.
- Create a System Administrator account by provisioning a corporate Information Technology expert with the Shared Services Administrator role. See "Provisioning Users and Groups" in the *Oracle Enterprise Performance Management System User Security Administration Guide*.

System Administrator

The System Administrator is typically a corporate Information Technology expert who has read, write, and execute access rights to all servers involved in an EPM System deployment.

Generally, the System Administrator performs these tasks:

- Disable the default EPM System Administrator account.
- Create at least one Functional Administrator.
- Set the security configuration for EPM System using the Shared Services Console.
- Optionally configure user directories as an external user directory.
- Monitor EPM System by periodically running the Log Analysis tool.

  The tasks that Functional Administrators perform are described in this guide.

Procedures to create a Functional Administrator:

- Configure the corporate directory as an external user directory. See Configuring User Directories.
- Provision a user or group with the required roles to create a Functional Administrator. See "Provisioning Users and Groups" in the *Oracle Enterprise Performance Management System User Security Administration Guide*.

  The Functional Administrator must be provisioned with these roles:

  - LCM Administrator role of Shared Services
  - Administrator and Provisioning Manager role of each deployed EPM System component

Functional Administrators

The Functional Administrator is a corporate user who is an EPM System expert. Typically, this user is defined in the corporate directory that is configured in Shared Services as an external user directory.

Functional Administrator performs EPM System administration tasks such as creating other Functional Administrators, setting up delegated administration, creating and provisioning applications and artifacts, and setting up EPM System auditing. The tasks that Functional Administrators perform are described in the *Oracle Enterprise Performance Management System User Security Administration Guide*.

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.

Launching Shared Services Console

You use a menu option in Oracle Hyperion Enterprise Performance Management Workspace to Access Oracle Hyperion Shared Services Console.

To launch the Shared Services Console:

1. Go to:

   `http://`*`web_server_name`*`:`*`port_number`*`/workspace`

   In the URL, *`web_server_name`* indicates the name of the computer where the web server used by Oracle Hyperion Foundation Services is running, and *`port_number`* indicates the web server port; for example, `http://myWebserver:19000/workspace`.

   > **Note:**
   >
   > If you are accessing EPM Workspace in secure environments, use `https` (not `http`) as the protocol and the secure web server port number. For example, use a URL such as: `https://myserver:19043/workspace`.

2. Click **Launch Application**.

   > **Note:**
   >
   > Pop-up blockers may prevent EPM Workspace from opening.

3. In **Logon**, enter your user name and password.

   Initially, the only user who can access Shared Services Console is the Oracle Enterprise Performance Management System Administrator whose user name and password were specified during the deployment process.

4. Click **Log On**.
5. Select **Navigate**, then **Administer**, and then **Shared Services Console**.

2

SSL-Enabling EPM System Components

Related Topics

- Assumptions
- Information Sources
- Location References
- About SSL-Enabling EPM System Products
- Supported SSL Scenarios
- Required Certificates
- Terminating SSL at the SSL Offloader
- Full SSL Deployment of EPM System
- Terminating SSL at the Web Server
- SSL for Essbase 11.1.2.4
- SSL for Essbase 21c

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.
- You are familiar with SSL concepts and procedures such as importing certificates.

  See Information Sources for a list of reference documents.

Information Sources

SSL-enabling Oracle Enterprise Performance Management 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.

  - **Sun Java System Directory Server:** See "Directory Server Security" in the *Sun Java System Directory Server Administration Guide.*
  - **Active Directory:** See Microsoft documentation.
- **Databases**: See the documentation from the database vendor.

Location References

This document refers to the following installation and deployment locations:

- *MIDDLEWARE_HOME* refers to the location of middleware components such as Oracle WebLogic Server, and, optionally, one or more *EPM_ORACLE_HOME*. The *MIDDLEWARE_HOME* is defined during Oracle Enterprise Performance Management 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. *EPM_ORACLE_HOME* resides within *MIDDLEWARE_HOME*. 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 Oracle Enterprise Performance Management System deployment process automatically deploys Oracle's EPM System products to work in both SSL and non-SSL modes.

Note:

- EPM System supports SSL over HTTP and JDBC only. It does not support other standards, for example Thrift and ODBC, for secure communication.
- To protect against the Poodle (Padding Oracle On Downgraded Legacy Encryption) vulnerability, which is an attack on the SSLv3 protocol, you must disable SSLv3 support in your servers and in the browsers that are used to access EPM System components. See your server and browser documentation for information to disable SSLv3 support.
- EPM System servers may fail to start if you disable non-SSL mode after configuring SSL.
  Enable secure replication for all EPM System servers in the domain to make them start when non-SSL mode is disabled.

While specifying common settings 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.

> **Note:**
>
> Redeploying your applications erases the custom application server and web server settings that you specify to enable SSL.

> **Note:**
>
> In Enterprise Performance Management System Release 11.2.x, Secure Sockets Layer (SSL) for MS SQL Server in the Repository Creation Utility (RCU) is not supported.

Supported SSL Scenarios

The following SSL scenarios are supported:

- SSL termination at the SSL offloader. See Terminating SSL at the SSL Offloader.
- Full SSL deployment. See Full SSL Deployment of EPM System.

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 Oracle Enterprise Performance Management 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 Oracle WebLogic Server.

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

- A root CA certificate

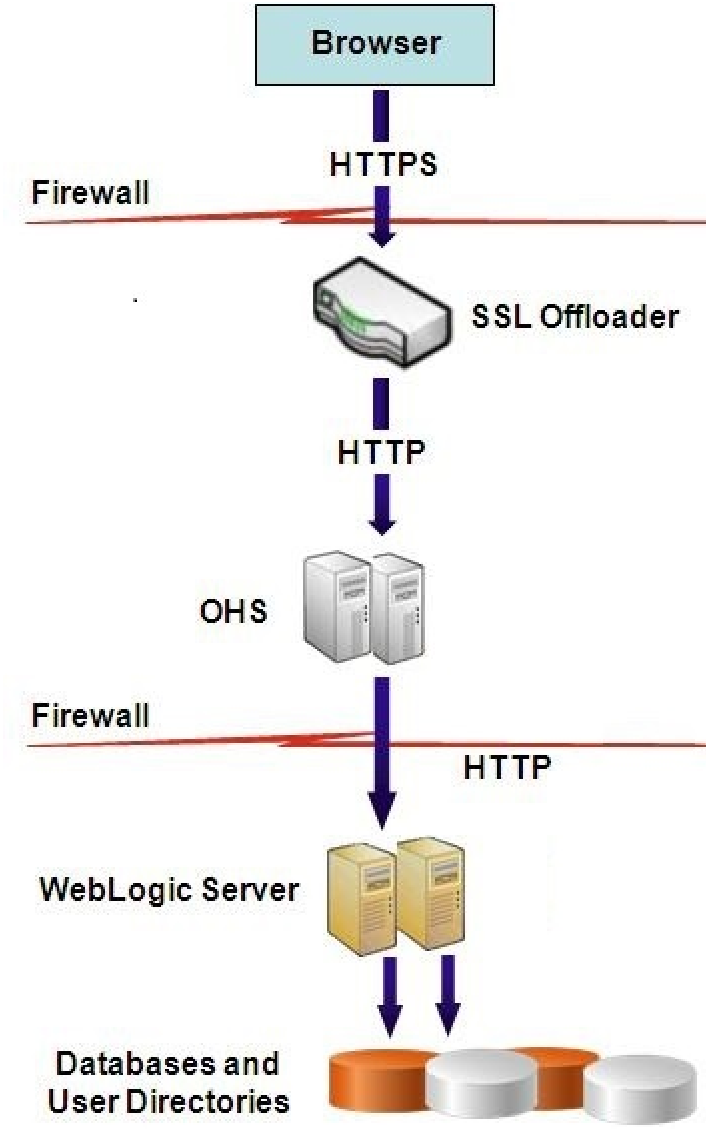
> **Note:**
>
> You need not install a root CA certificate in the Java keystore if you are using certificates from a well-known third-party CA whose root certificate is already installed in the Java keystore.
>
> Firefox and Internet Explorer are preloaded with certificates of well-known third-party 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. If you are acting as your own CA, web clients cannot establish an SSL handshake with the server if your CA root certificate is not available to the browser from which the client 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

Deployment Architecture

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



Assumptions

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.

When SSL is being terminated at Oracle HTTP Server (OHS) or load balancer, you must:

- Set every Logical Web Application to non-ssl virtual host of load balancer or Oracle HTTP Server (for example, `empinternal.myCompany.com:80` where 80 is the non-SSL port). Open Configuration screen, complete these steps:
    1. Expand **Hyperion Foundation** configuration task.
    2. Select **Configure Logical Address for Web Applications**.
    3. Specify the *Host name*, non-SSL port number and SSL port number.

- Set external URL to SSL-enabled virtual host of load balancer or Oracle HTTP Server (for example, `empexternal.myCompany.com:443` where 443 is the SSL port). Open Configuration screen, complete these steps:
  1. Expand **Hyperion Foundation** configuration task.
  2. Select **Configure Common Settings**.
  3. Select **Enable SSL offloading** under External URL Details.
  4. Specify the *External URL Host* and *External URL Port*.

> **Note:**
>
> Redeploying web applications or reconfiguring the web server using **configtool** will replace the settings for Logical Web Application and external URLs.

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 address for 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

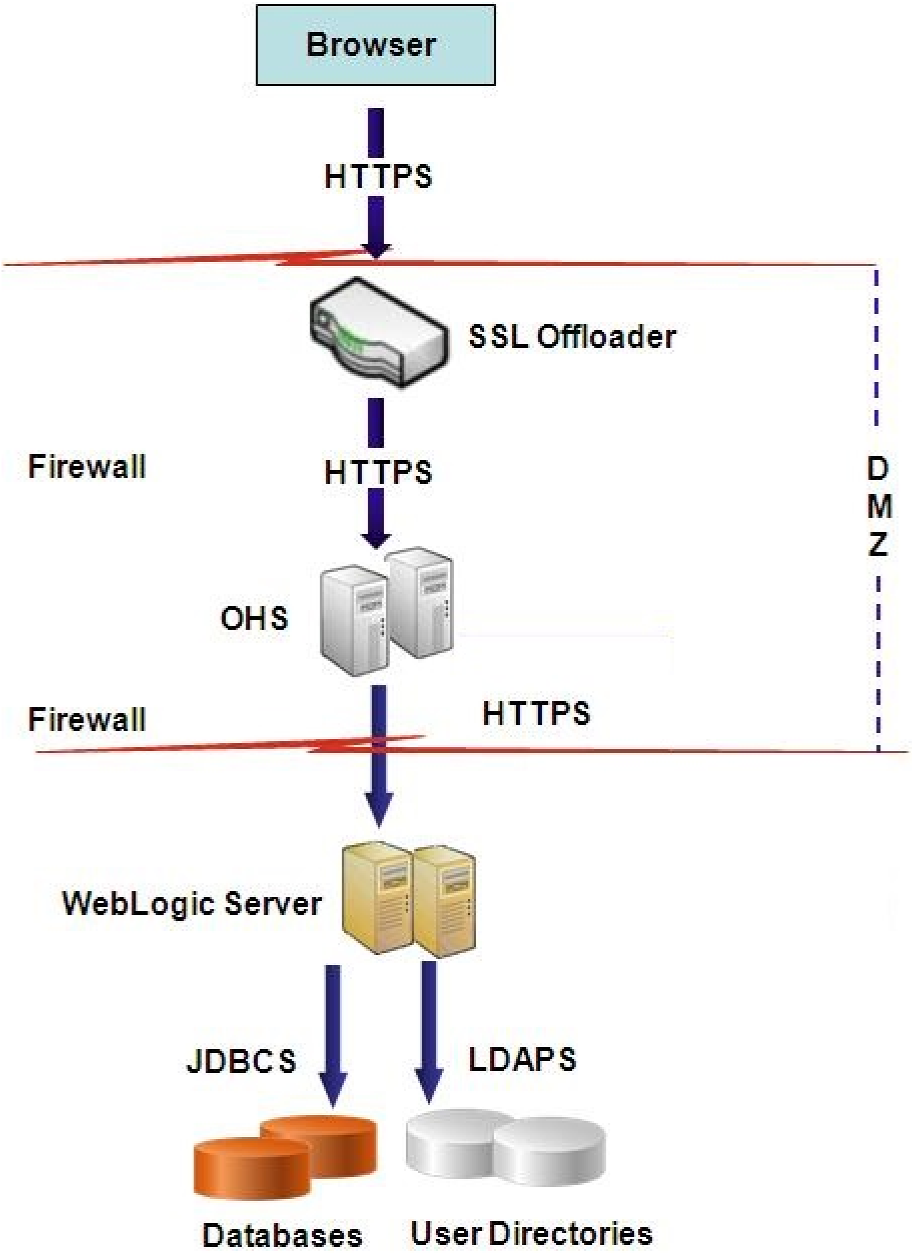
Related Topics

- 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 Oracle Enterprise Performance Management System deployment scenario is the most secure.

The illustrated concept:



Assumptions

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

Oracle Enterprise Performance Management System components, including Oracle 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 on 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=TCPS)(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 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

Related Topics

- Reconfiguring EPM System Common Settings
- Optional: Installing Root CA Certificate for WebLogic Server
- Installing Certificate on the WebLogic Server
- Configuring WebLogic Server
- Enabling HFM Server Connection with an SSL-Enabled Oracle Database
- Oracle HTTP Server Procedures
- Configuring EPM System Web Components Deployed on WebLogic Server
- Update the Domain Configuration
- Restarting Servers and EPM System
- Testing the Deployment

- Configuring SSL-Enabled External User Directories

Reconfiguring EPM System Common Settings

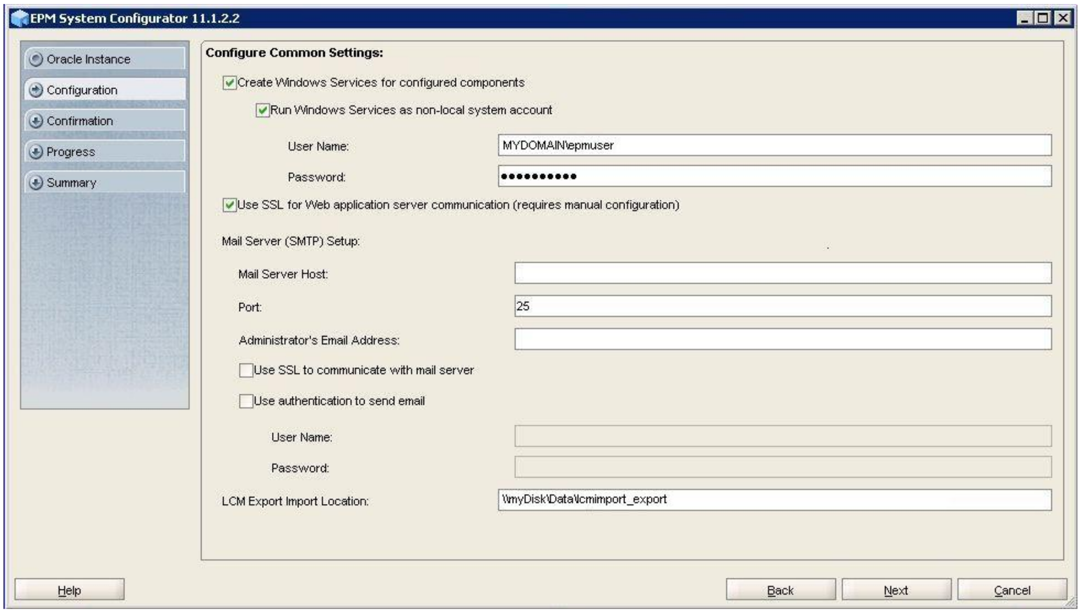
During this process, you select the settings that force Oracle Enterprise Performance Management System components to use SSL communication.

> **Note:**
>
> **If you are SSL-enabling the Oracle Hyperion Financial Management web server:** Before configuring Financial Management, you must make the cookie secure by editing the session-descriptor of HFM WebApp in `weblogic.xml`.
>
> 1. Expand the Financial Management web archive using a tool such as 7 Zip. Location of `weblogic.xml` in the archive is *`EPM_ORACLE_HOME`*`\products\FinancialManagement\AppServer\InstallableApps\HFMWebApplication.ear\HFMWeb.war\WEB-INF\weblogic.xml`.
> 2. Include the following directive in the session-descriptor of HFM WebApp in `weblogic.xml`:
>   `<cookie-secure>true</cookie-secure>`
> 3. Save `weblogic.xml`
> 4. Click **Yes** when 7 Zip queries whether you want to update the archive.

To reconfigure EPM System for SSL:

1. Launch EPM System Configurator.
2. On **Select the EPM Oracle Instance to which the configuration will be applied**, complete these steps:
   a. In **EPM Oracle instance name**, enter the instance name that you used while originally configuring EPM System components.
   b. Click **Next**.
3. On the Configuration screen, complete these steps:
   a. Clear **Uncheck All**.
   b. Expand **Hyperion Foundation** configuration task, and then select **Configure Common Settings**.
   c. Click **Next**.
4. In **Configure Common Settings**, complete these steps:

   > **Caution:**
   >
   > Before selecting the settings to use SSL to communicate with the email server, ensure that the email server is configured for SSL.



Configure Common Settings:

☑ Create Windows Services for configured components

☑ Run Windows Services as non-local system account

User Name: MYDOMAIN\epmuser

Password: ●●●●●●●●●●

☑ Use SSL for Web application server communication (requires manual configuration)

Mail Server (SMTP) Setup:

Mail Server Host:

Port: 25

Administrator's Email Address:

□ Use SSL to communicate with mail server

□ Use authentication to send email

User Name:

Password:

LCM Export Import Location: \\myDisk\Data\lcmimport_export

a. Select **Use SSL for Java 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.

e. Click **Next**.

5. Click **Next** on subsequent EPM System Configurator screens.

6. When the deployment process is complete, the Summary screen is displayed. 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 JVM keystore. Complete the procedures in this section if you are not using certificates from a well-known third-party CA (not recommended). Default JVM keystore location is `MIDDLEWARE_HOME/jdk/jre/lib/security/cacerts`.

> **Note:**
>
> Perform this procedure on each Oracle Enterprise Performance Management System server.

To install the root CA certificate:

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

2. From a console, change directory to `MIDDLEWARE_HOME/jdk/jre/bin`.

3. Execute a keytool command such as the following to install the root CA certificate into the 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 JVM keystore with `Blister` as the certificate alias in the keystore. Storepass `example_pwd` is assumed.

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

Installing Certificate on the WebLogic Server

The default Oracle 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 Oracle Enterprise Performance Management System web components.

To create a custom keystore and import certificate:

1. From a console, change directory to `MIDDLEWARE_HOME/jdk/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 Oracle Enterprise Performance Management System web components, you must configure them for SSL communication.

To configure the web components for SSL:

1. Start the Oracle WebLogic Server by executing `MIDDLEWARE_HOME/user_projects/domains/EPMSystem/bin/startWebLogic.cmd`:
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 Oracle Hyperion Foundation Services components, you work with the `FoundationService0` 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**:
    - 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.
    - In **Custom Trust Keystore Type**, enter `jks`.
    - In **Custom Trust Keystore Passphrase** and **Confirm Custom Trust Keystore Passphrase**, enter the keystore password.
18. Click **Save**.
19. Specify SSL settings:
    - Select **SSL**.
    - In **Private Key Alias**, enter the alias that you specified while importing the signed WebLogic Server certificate.
    - In **Private Key Passphrase** and **Confirm Private Key Passphrase**, enter the password to be used to retrieve the private key.
    - Click **Save**.

> **Note:**
>
> If you are using SHA-2 certificates, you must select **Use JSSE SSL** setting for every managed server that is used to support EPM System. This setting is available on the Advanced tab of SSL page. You need to restart WebLogic Server to activate this change.

20. Enable secure replication for the server:
    a. In the left pane of the console, expand **Environment** and then click **Clusters**.
    b. In Summary of Clusters, click the name of the server, for example `Foundation Services`, for which you want to enable secure replication.

       The Configuration tab of the Settings screen for the selected server is displayed.
    c. Click **Replication** to open the Replication tab.
    d. Select **Secure Replication Enabled**. You may need to click **Lock & Edit** before you can select this option.
    e. Click **Save**.
21. Complete step 6 through step 20 for each managed server belonging to this host.
22. Enable secure replication to provide channel for replication calls for cluster.

    See Oracle metalink document 1319381.1 for details.
    - In the Administration Console, expand **Environment**, and then select **Clusters**.
    - Select **Replication**.

   - On **Replication**, select (check) **Secure Replication Enabled**.
   - Click **Save**.

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

Enabling HFM Server Connection with an SSL-Enabled Oracle Database

The network connection between the HFM DataSource and the Oracle database can be encrypted using SSL. For this to work, the Oracle Wallet must be configured as outlined in the Oracle documentation. The TNS Listener must also be configured to listen on a new port for SSL encrypted connections. Finally, the appropriate certificates need to be loaded into the keystore and truststore on the servers hosting the HFM DataSource. The instructions below are referred from the Oracle Database documentation.

Prerequisites

Ensure that the folloiwng prerequisites are met before proceeding with the steps below:

- A functioning database server.
- Ensure that no local or network firewalls are blocking any communication with the server on port where the SSL enabled TNS listener is running.

In the examples below, Oracle 12c (12.1.0.2) version running on MS Windows Server 2016 has been used. These instructions will work equally well on a Linux installation provided that the paths specified are for the wallet files are Linux filesystem paths and the environment variable substitutions are properly changed for the shell being used on the database server. These same instructions have been successfully used on 19c development and support instances.

The examples in this article use self signed certificates, but you can also use proper certificate authority certificates if you prefer. See Oracle Database documentation for the exact steps to follow when installing a certificate issued by a certificate authority.

Configuring Oracle Database

To configure the Oracle Database, follow the steps below:

1. Create a new auto-login wallet on the database server.

   > **Note:**
   >
   > These steps are required only if an Oracle Wallet has not been created previously. The following steps are not necessary if the GUI Oracle Wallet tool is used on the database server.

   ```
   C:\> cd %ORACLE_HOME%
   C:\oracledb\12.1.0\home> mkdir wallet
   C:\oracledb\12.1.0\home> orapki wallet create -wallet wallet -pwd
   password1 -auto_login
   ```

   You may ignore any messages that prompt you to use `-auto_login_local` on the `orapki` command line. If you run into SSL authentication failure error, see Doc ID 2238096.1 to troubleshoot the issue.

Also, check the security permission of the file `cwallet.sso` (under the wallet directory) and ensure the Oracle listener service user has read permission to this file. Without read permission, the SSL handshake will fail later on. This situation will occur if the Oracle database was installed with the suggested Oracle user who is not allowed to log on. If the Oracle database was installed with the Oracle user, then the TNS Listener must be run as a differnt user.

2. Create a self-signed certificate and load it into the wallet

   ```
   C:\oracledb\12.1.0\home> orapki wallet add -wallet wallet -pwd password1 -
   dn "CN={FQDN of db server}" -
   keysize 1024 -self_signed -validity 3650
   ```

   The password `password1` in the example above must match the password specified in *Step 1*.

3. Export the newly created self-signed certificate

   ```
   C:\oracledb\12.1.0\home> orapki wallet export -wallet wallet -pwd
   password1 -dn "CN={FQDN of db server}"
   -cert %COMPUTERNAME%-certificate.crt
   ```

4. Copy the exported Base64 certificate file to the HFM server(s).
5. Configure the SQL*NET and the TNS Listeners:
   a. Identify an unused port on the database server. The example below creates the new listener on port `1522`. The typical port used for SSL connections is `2484` and you may use any available port. You must check that the port you want to use is available on the database server before proceeding and adjust as necessary.
   b. Update `SQLNET.ORA`. The DIRECTORY element of the WALLET_LOCATION declaration must point to the wallet created in *Step 1* above.

      ```
      SQLNET.AUTHENTICATION_SERVICES= (TCPS, NTS, BEQ)
      NAMES.DIRECTORY_PATH= (TNSNAMES, EZCONNECT)
      WALLET_LOCATION=
      (SOURCE =
      (METHOD = FILE)
      (METHOD_DATA =
      (DIRECTORY = C:\oracledb\12.1.0\home\wallet)
      )
      )
      SSL_CLIENT_AUTHENTICATION = FALSE
      ```

   c. Update LISTENER.ORA to define a new listener. Use the port that was identified in *Step 5a* above.

      ```
      SID_LIST_LISTENER =
      (SID_LIST =
      (SID_DESC =
      (SID_NAME = CLRExtProc)
      (ORACLE_HOME = C:\oracledb\12.1.0\home)
      (PROGRAM = extproc)
      (ENVS = "EXTPROC_DLLS=ONLY:C:\oracledb\12.1.0\home\bin\oraclr12.dll")
      )
      )
      ```

```
SSL_CLIENT_AUTHENTICATION = FALSE
WALLET_LOCATION=
(SOURCE =
(METHOD = FILE)
(METHOD_DATA =
(DIRECTORY = C:\oracledb\12.1.0\home\wallet)
)
)
LISTENER =
(DESCRIPTION_LIST =
(DESCRIPTION =
(ADDRESS = (PROTOCOL = TCP)(HOST = myServer)(PORT = 1521))
)
(DESCRIPTION =
(ADDRESS = (PROTOCOL = TCPS)(HOST = myServer)(PORT = 1522))
)
(DESCRIPTION =
(ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC1521))
)
)
ADR_BASE_LISTENER = C:\oracledb
```

d. Create a new entry in `TNSNAMES.ORA` for the new port.

```
ORCL_SSL =
(DESCRIPTION =
(ADDRESS = (PROTOCOL = TCPS)(HOST = myServer)(PORT = 1522))
(CONNECT_DATA =
(SERVER = DEDICATED)
(SERVICE_NAME = myServer_service)
)
)
```

You must specify the same port that was identified in *Step 5a* above and used in *Step 5c*.

e. Restart the TNS Listener.

```
C:\oracledb\12.1.0\home>lsnrctl stop
C:\oracledb\12.1.0\home>lsnrctl start
```

f. Verify that the new TNS listener is working

```
C:\oracledb\12.1.0\home>tnsping orcl_ssl
TNS Ping Utility for 64-bit Windows: Version 12.1.0.2.0 - Production on
10-SEP-2019 15:43:22
Copyright (c) 1997, 2014, Oracle. All rights reserved.
Used parameter files:
C:\oracledb\12.1.0\home\network\admin\sqlnet.ora
Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS = (PROTOCOL = TCPS)(HOST
= myServer)
(PORT = 1522)) (CONNECT_DATA = (SERVER = DEDICATED) (SERVICE_NAME =
myServer_service)))
OK (130 msec)
```

Configuring HFM server to use SSL database connections

Adding the database's certificate to the truststore on the HFM server(s)

The following steps must be performed on each and every EPM server where the HFM datasource runs. The `%MW_HOME%` environment variable used below is the location of the Oracle Middleware installation. This environment variable is not created by default during the EPM installation and is used here to show the parent directory of the EPM installation.

The location of the EPM installation is specified by the `EMP_ORACLE_HOME` environment variable. The example below places the keystore and truststores in a directory co-located with the EPM installation. The keystore and truststore files can be located anywhere on the HFM server's filesystem.

1. Create a new directory under `%MW_HOME%` to store the Java keystore and PKCS12 truststore.
   a. cd `%MW_HOME%`
   b. mkdir `certs`
2. Copy the Java keystore file cacerts from the JDK.
   a. cd `%MW_HOME%\certs`
   b. copy `%MW_HOME%\jdk1.8.0_181\jre\lib\security\cacerts testing_cacerts`
      The reason for copying the JDK's keystore and using it instead of the JDK's default keystore is that if the JDK is upgraded and the previous JDK gets deleted, the keys and certificated inserted into the default keystore will be lost.
3. Copy the Base 64 certificate to `%MW_HOME%\certs`.
4. Import the certificate into the Java keystore file `testing_cacerts`.
   a. For example, `keytool -importcert -file bur00cbb-certificate.crt -keystore testing_cacerts -alias "myserver"`
      i. You will have to specify the password for the keystore.
      ii. You should replace "myserver" with the fully qualified domain of the database server.
   b. When you are prompted with a question of whether or not the certificate should be trusted, specify **y**.
5. Create the truststore in PKCS12 format from the JDK's Java keystore file. For example,

```
keytool -importkeystore -srckeystore testing_cacerts -srcstoretype JKS -
deststoretype PKCS12 -destkeystore testing_cacerts.pfx
```

Updating the HFM JDBC connections to use SSL

1. Reconfigure the HFM database JDBC connection to use SSL.
   a. Launch the EPM Configuration tool.
      i. Select the **Configure Database** and **Deploy to Application Server** nodes under the **Financial Management** node.
      ii. Click **Next**.
      iii. Perform each of these steps for the HFM JDBC connection

i. Enter the SSL port, Service name, username and password for the connection in the port, service name, username,and password columns.

ii. Click **(+)** to open the **Advanced database options.**

iii. Select **Use secure connections** checkbox.

iv. Enter the location of the Java keystore created in *Step 2*.

v. Click **Apply**.

vi. Click **(+)** to open the **Advanced database options.**

vii. Click **Edit and use modified JDBC URL**. Note that no changes should be made to the displayed JDBC URL.

viii. Click **Apply**.

ix. Click **Next**.

b. Go through through the remaining steps to deploy the HFM application as described in the EPM documentation.

2. Open a command window or shell to manually update the EPM registry so that the ODBC connection used by the DataSource can be SSL enabled.
Execute each of the commands listed below:

```
epmsys_registry.bat addproperty FINANCIAL_MANAGEMENT_PRODUCT/DATABASE_CONN/
@ODBC_TRUSTSTORE "C:
\Oracle\Middleware\certs\testing_cacerts.pfx"
epmsys_registry.bat addencryptedproperty FINANCIAL_MANAGEMENT_PRODUCT/
DATABASE_CONN
/@ODBC_TRUSTSTOREPASSWORD <truststorepassword>
epmsys_registry.bat addproperty FINANCIAL_MANAGEMENT_PRODUCT/DATABASE_CONN
/@ODBC_VALIDATESERVERCERTIFICATE false
```

In the above examples, the path `C:\Oracle\Middleware` is the value of `%MW_HOME%` in steps 1,2, and 3.

The property `FINANCIAL_MANAGEMENT_PRODUCT/DATABASE_CONN/@ODBC_VALIDATESERVERCERTIFICATE` should only be set to false if a self-signed certificate is being used. The value of the `FINANCIAL_MANAGEMENT_PRODUCT/DATABASE_CONN/@ODBC_TRUSTSTOREPASSWORD` should be the password of the original Java keystore copied in *Step 2*.

Update the TNS names entry used by HFM

Edit `TNSNAMES.ORA` to create a new entry and rename the old entry. The following example shows an updated `TNSNAMES.ORA` file on the HFM server that has the necessary changes applied. The reason for these changes is that HFM looks for and uses a TNS names entry named `HFMTNS`. This entry must have the protocol and the port changed for `XFMDataSource` to work properly.

```
HFMTNS_UNENC =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP)(HOST = myserver)(PORT = 1521))
)
(CONNECT_DATA =
(SERVICE_NAME = myserver_service)
(SERVER = DEDICATED)
)
)
HFMTNS =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCPS)(HOST = myserver)(PORT = 1522))
)
(CONNECT_DATA =
(SERVICE_NAME = myserver_service)
(SERVER = DEDICATED)
)
)
```

The original `HFMTNS` entry has been renamed `HFMTNS_UNENC`. The new `HFMTNS` was made by copying the `HFMTNS_UNENC` entry, renaming it `HFMTNS`. The protocol was then updated to TCPS and the port was changed to `1522`. The port specified must be the same port specified in the `TNS LISTENER.ORA` file.

Oracle HTTP Server Procedures

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.

Note: Starting 11.2.x, Oracle Wallet Manager is not installed with Oracle HTTP Server. The Oracle Wallet Manager gets installed only if you install the Oracle Database Client. You must use the wallet manager available with Database Client to create the wallet and import the certificate. If you are configuring Oracle HTTP Server for SSL, ensure that you always install the Oracle Database Client 64-bit as part of the installation of your EPM system products.

To create and install Oracle HTTP Server certificate :

1. On each machine that hosts Oracle HTTP Server, launch the Wallet Manager.

   Select **Start**, then **All Programs**, **Oracle-OHxxxxxx**, then **Integrated Management Tools**, and then **Wallet Manager**.

   `xxxxxx` is the Oracle HTTP Server instance number.

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.
   a. In Oracle Wallet Manager, right-click **Certificate: [Empty]** and select **Add 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.

Setting Up Oracle Wallet Using ORAPKI (on Linux)

To set up Oracle Wallet using ORAPKI command line, complete the following steps:

1. Create a folder for your wallet:

```
$ mkdir /MIDDLEWARE_HOME/oracle_common/wallet
```

2. Add the location of the orapki utility to your path:

```
$ export PATH=$PATH:$MIDDLEWARE_HOME/oracle_common/bin
```

3. Create a wallet to hold your certificate:

```
>$ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet create -wallet
[wallet_location] -auto_login
```

   This command prompts you to enter and reenter a wallet password, if no password has been specified on the command line. It creates a wallet in the location specified for –`wallet`.

4. Generate a certificate signing request (CSR) and add it to your wallet:

   ```
   $ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet add -wallet
   [wallet_location] -dn 'CN=<CommonName>,OU=<OrganizationUnit>, O=<Company>,
   L=<Location>, ST=<State>, C=<Country>' -keysize 512|1024|2048|4096 -pwd
   [Wallet_Password]
   ```

5. Add the root and intermediate certificate into the trusted keystore

   ```
   $ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet add -wallet
   [wallet_location] -trusted_cert -cert [certificate_location] [-pwd]
   ```

6. Use your CA (Cerificate Authority) to sign the CSR (Certificate Signing Request). To export the certtificate request from an Oracle Wallet:

   ```
   $ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet export -wallet
   [wallet_location] -dn 'CN=<CommonName>,OU=<OrganizationUnit>, O=<Company>,
   L=<Location>, ST=<State>, C=<Country>' -request
   [certificate_request_filename] [-pwd]
   ```

7. Import the signed CSR into the wallet:

   ```

       $ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet add -wallet
   [wallet_location] -user_cert -cert [certificate_location] [-pwd]
   ```

8. To display the contents of the wallet:

   ```
   $ MIDDLEWARE_HOME/oracle_common/bin/orapki wallet display -wallet
   [wallet_location] [-pwd]
   ```

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.
2. Start EPM System Configurator for the instance.
3. In the configuration task selection screen, complete these steps, and then click **Next**.
   a. Clear the selection from **Uncheck All**.
   b. Expand **Hyperion Foundation** task group, and then select **Configure Web Server**.
4. In **Configure Web Server**, click **Next**.
5. In **Confirmation**, click **Next**.
6. In **Summary**, click **Finish**.
7. Using a text editor, open *EPM_ORACLE_INSTANCE*`/httpConfig/ohs/config/fmwconfig/components/OHS/ohs_component/ssl.conf`.

8. Ensure that the SSL port you are using is listed under `OHS Listen port`. similar to the following:

   If you are using `19443` as the SSL communication port, your entries should be as follows:

   ```
   Listen 19443
   ```

9. Set `SSLSessionCache` parameter value to `none`.

10. Update the configuration settings of each Oracle HTTP Server in your deployment.

    a. Using a text editor, open *EPM_ORACLE_INSTANCE*`/httpConfig//ohs/config/fmwconfig/components/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`.

11. 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/fmwconfig/components/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. Set the value of `SecureProxy` directive is set to `ON`.

    ```
    SecureProxy ON
    ```

    d. Ensure that the `LocationMatch` definitions for deployed Oracle Enterprise Performance Management System components are similar to the following Oracle Hyperion Shared Services example, which assumes a Oracle WebLogic Server cluster (on `myserver1` and `myserver2` using SSL port 28443):

    ```
    <LocationMatch /interop/>
        SetHandler weblogic-handler
        pathTrim /
        WeblogicCluster myServer1:28443,myServer2:28443
        WLProxySSL ON
    </LocationMatch>
    ```

    e. Save and close `mod_wl_ohs.conf`.

Configuring EPM System Web Components Deployed on WebLogic Server

After deploying Oracle Enterprise Performance Management System web components, you must configure them for SSL communication.

To configure the web components for SSL:

1. Start the Oracle WebLogic Server by executing a file stored in *`EPM_ORACLE_INSTANCE`*`/domains/EPMSystem/bin/startWebLogic.cmd`:
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 user name and password to access the `EPMSystem`. The user name and password are specified in EPM System Configurator during the configuration process.
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 you want to SSL-enable.

   For example, if you installed all Oracle Hyperion Foundation Services components, you can SSL-enable these servers:

   - `CalcManager`
   - `FoundationServices`
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 WebLogic Server SSL listen port.
10. Specify the identity and trust keystores to use.

    - Select **Keystores** to open the Keystores tab.
    - In **Keystores**, select an option:

    a. Select **Keystores** to open the Keystores tab.

    b. In **Keystores**, 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

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

    d. In **Custom Identity Keystore Type**, enter `jks`.

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

    f. If you selected **Custom Identity and Custom Trust** in **Keystores**:

    - 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.
    - In **Custom Trust Keystore Type**, enter `jks`.
    - In **Custom Trust Keystore Passphrase** and **Confirm Custom Trust Keystore Passphrase**, enter the keystore password.

  g. Click **Save**.

11. Specify SSL settings.
    - Select **SSL**.
    - In **Private Key Alias**, enter the alias that you specified while importing the signed WebLogic Server certificate.
    - In **Private Key Passphrase** and **Confirm Private Key Passphrase**, enter the password to be used to retrieve the private key.
    - **Oracle Hyperion Provider Services web application only:** If you are using wildcard certificates to encrypt communication between WebLogic Server and other EPM System server components, disable host name verification for Provider Services web application.
      – Select **Advanced**.
      – In **Hostname Verification**, select **None**.
    - Click **Save**.
12. In **Change Center**, click **Activate Changes**.

Update the Domain Configuration

This process updates the domain configuration. Create a complete backup of your deployment before starting this procedure. Oracle recommends that you test this procedure on a test deployment before making changes on a production deployment.

To update the domain configuration:

1. Navigate to `MIDDLEWARE_HOME/oracle_common/bin directory` directory:
   ```
   cd MIDDLEWARE_HOME/oracle_common/bin
   ```
2. Set *`ORACLE_HOME`*, *`WL_HOME`* and *`JAVA_HOME`*.
   ```
   set ORACLE_HOME= /Oracle/Middleware

   set WL_HOME= /Oracle/Middleware/wlserver

   set JAVA_HOME= /Oracle/Middleware/jdk
   ```
3. In Web Logic Console, enable the HTTP port for the Admin server.
4. Create a keystore using a command similar to the following:
   ```
   libovdconfig.bat -host HOSTNAME -port 7001 -userName USERNAME -domainPath
   %MWH%\user_projects\domains\EPMSystem -createKeystore
   ```
   In this command, replace the *`HOSTNAME`* and *`USERNAME`* with the host name of the Web Logic server and the user name of the Administrator respectively. Make sure that output reports the successful creation of the OVD keystore.
5. Export the SSL certificate from AdminServer.

> **Note:**
>
> This step is applicable only for Embedded LDAP (Default Authenticator). For other LDAP's, the certificate must be exported by using the appropriate LDAPspecific commands. The certificate file format mut be **Base 64 Encoded x.509**

a. Using Internet Explorer access the Web Logic admin console by connecting to `https://HOSTNAME:7002/console`

b. Click **View Certificate**, then **Details**, and select **Copy to file** to export the SSL certificate.

c. Save the certificate as a **Base 64 Encoded x.509** certificate file to a local directory; for example, as `C:\certificate\slc17rby.cer`.

d. Move the certificate to the server.

6. Using keytool, import the certificate into the keystore that you created in step 4. Use commands similar to the following (assuming that the *JAVA_HOME* (and keytool executable) is in the path:

```
export PATH=$JAVA_HOME/bin:$PATH
```

```
keytool -importcert -keystore
DOMAIN_HOME\config\fmwconfig\ovd\default\keystores/adapters.jks -storepass
PASSWORD -alias wcp_ssl -file CERTIFICATE_PATH -noprompt
```
, for example:

```
keytool -importcert -keystore %MWH%
\user_projects\domains\EPMSystem\config\fmwconfig\ovd\default\keystores/
adapters.jks -storepass examplePWD -alias wcp_ssl -file
C:\certificate\slc17rby.cer -noprompt
```

> **Note:**
>
> - The password used in this command must match the password used while generating the keystore in step 4.
> - *CERTIFICATE_PATH* is the location and name of the certificate
> - alias can be any alias of your choice.

On successfully importing the certificate, keytool displays the message `Certificate was added to keystore`.

7. In Web Logic Console, enable SSL port for the Admin Server in addition to the HTTP port.
8. Restart Weblogic Admin Server and Managed Servers.
9. Log into Oracle Hyperion Enterprise Performance Management Workspace using a secure connection to verify that everything is working.

Restarting Servers and EPM System

Restart all servers in the deployment, and then start Oracle Enterprise Performance Management 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 Oracle Hyperion Enterprise Performance Management 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 Oracle Enterprise Performance Management System components.

Configuring SSL-Enabled External User Directories

Assumptions

- The external user directories that you plan to configure in Oracle Hyperion 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, then you must import the root certificate of the CA that signed the server certificate into the following keystores:

> **Note:**
>
> During application deployment, WebLogic adds the `-Djavax.net.ssl.trustStore` directive pointing to `DemoTrust.jks` in `setDomainEnv.sh` or `setDomainEnv.cmd`. Remove `-Djavax.net.ssl.trustStore` from `setDomainEnv.sh` or `setDomainEnv.cmd` if you are not using the default WebLogic certificate.

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

- All Oracle Enterprise Performance Management System servers:

  JVM keystore: *`MIDDLEWARE_HOME`*`/jdk/jre/lib/security/cacerts`

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

  `MIDDLEWARE_HOME``/jdk/jre/lib/security/cacerts`

Configure External User Directories

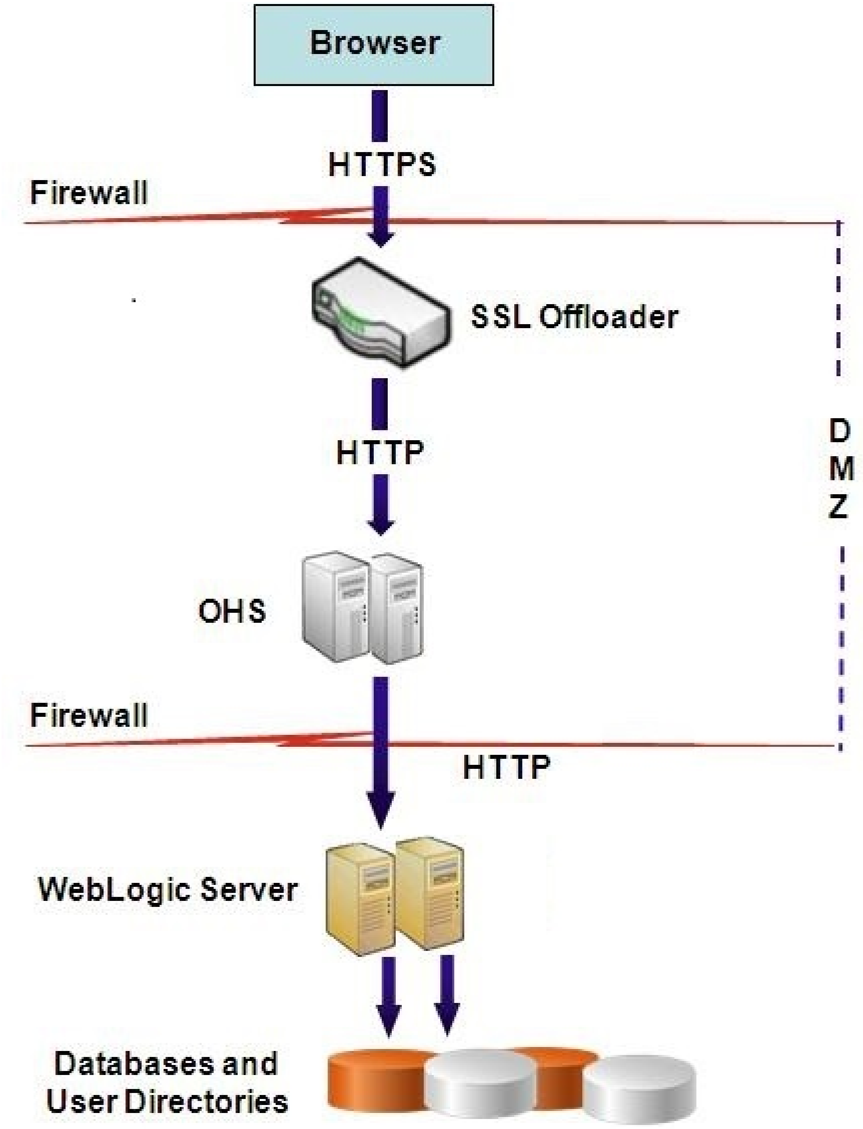
You configure user directories using the Shared Services Console. While configuring user directories, you must select the `SSL Enabled` option that instructs EPM System security to use the secure protocol to communicate with the user directory. You can SSL-enable a connection between EPM System security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

See "Configuring User Directories" in the *Oracle Enterprise Performance Management System User Security Administration Guide*.

Terminating SSL at the Web Server

Deployment Architecture

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



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.

When SSL is being terminated at Oracle HTTP Server (OHS) or load balancer, you must:

- Set every Logical Web Application to non-ssl virtual host of load balancer or Oracle HTTP Server (for example, `empinternal.myCompany.com:80` where 80 is the non-SSL port). Open Configuration screen, complete these steps:
  1. Expand **Hyperion Foundation** configuration task.
  2. Select **Configure Logical Address for Web Applications**.
  3. Specify the *Host name*, non-SSL port number and SSL port number.
- Set external URL to SSL-enabled virtual host of load balancer or Oracle HTTP Server (for example, `empexternal.myCompany.com:443` where 443 is the SSL port). Open Configuration screen, complete these steps:
  1. Expand **Hyperion Foundation** configuration task.
  2. Select **Configure Common Settings**.
  3. Select **Enable SSL offloading** under External URL Details.
  4. Specify the *External URL Host* and *External URL Port*.

> **Note:**
>
> Redeploying web applications or reconfiguring the web server using **configtool** will replace the settings for Logical Web Application and external URLs.

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. 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*

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.

SSL for Essbase 11.1.2.4

Overview

This section explains the procedures for replacing the default certificates that are used to secure communication between an Oracle Essbase instance and components such as MaxL, Oracle Essbase Administration Services Server, Oracle Essbase Studio Server, Oracle Hyperion Provider Services, Oracle Hyperion Foundation Services, Oracle Hyperion Planning, Oracle Hyperion Financial Management, and Oracle Hyperion Shared Services Registry.

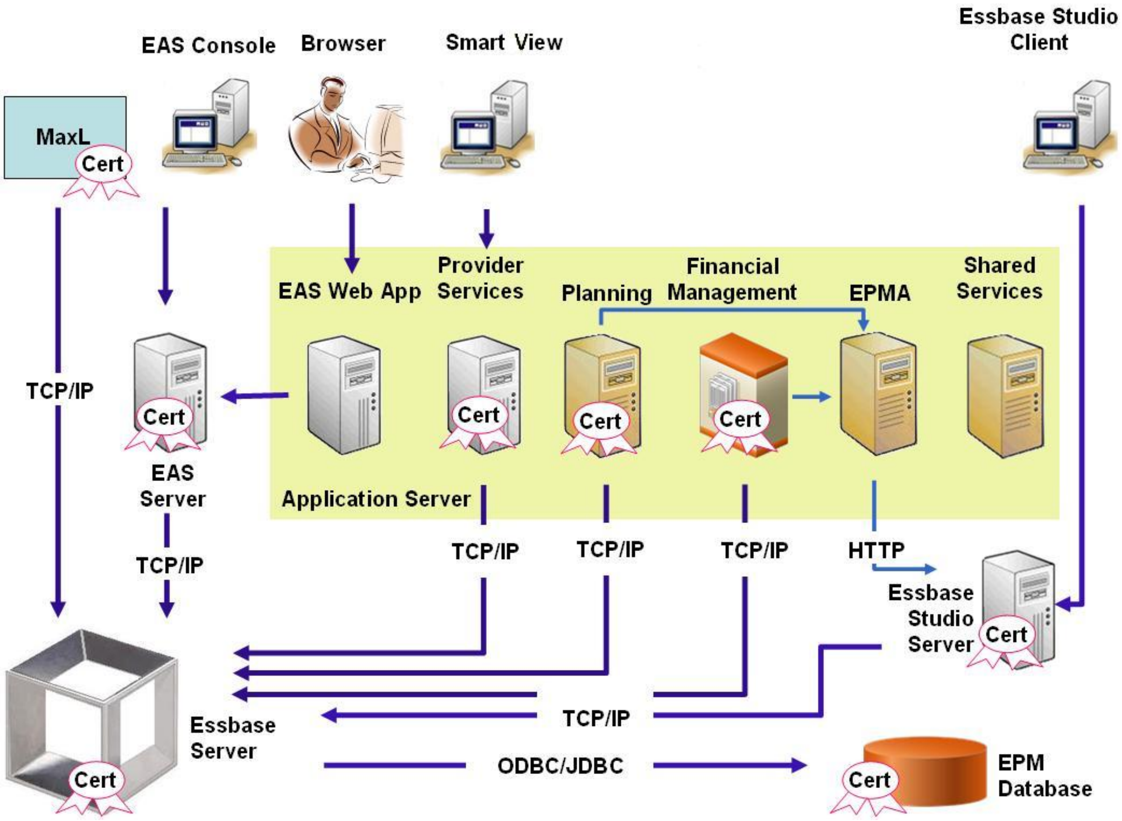
Default Deployment

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.

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 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.

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 to establish a connection to Essbase require that the root CA certificate be stored in an Oracle Wallet. Components that use JAPI to establish a connection require that the root CA certificate be stored in a Java keystore. The required certificates and their locations 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 2-1 Required Certificates and Their Locations

Component[1]	Keystore	Certificate [2]
MaxL	Oracle Wallet	Root CA certificate
Administration Services Server	Oracle Wallet	Root CA certificate
Provider Services	Oracle Wallet	Root CA certificate
Oracle Enterprise Performance Management System Database	Oracle Wallet	Root CA certificate
Essbase Studio Server	Java Keystore	Root CA certificate
Planning	• Oracle Wallet • Java Keystore	Root CA certificate
Financial Management	Java Keystore	Root CA certificate
Essbase (Server and Agent) [3]	• Oracle Wallet • Java Keystore	• Root CA certificate • Signed certificate for Essbase Server and Agent
Oracle Hyperion Shared Services Repository		

[1] You need only one instance of the keystore to support multiple components that use a 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 in the Java keystore.

Installing and Deploying Essbase Components

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

The EPM 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

Creating Certificate Requests and Obtaining Certificates

Generate a certificate request to obtain a certificate for the server that hosts Oracle 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 for 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\EPMSystem11R1\Essbase_ssl\keystore -storepass password
```

Obtaining and Installing Root CA Certificate

The root CA certificate verifies 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 .

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

Oracle Wallet

Refer to Required Certificates and Their Location 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 the root CA certificate.

Java Keystore

Refer to Required Certificates and Their Location 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 that 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 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 for 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
```

Update Essbase Server Registry Values

Windows

1. In a command prompt, change directory to `EPM_ORACLE_INSTANCE/epmsystem1/bin`.
2. Run these commands to update Windows Registry:

   ```
   epmsys_registry.bat updateproperty "#<Object ID>/@EnableSecureMode" true
   ```

   ```
   epmsys_registry.bat updateproperty "#<Object ID>/@EnableClearMode" false
   ```

   Be sure to replace `<Object ID>` with the Essbase Server component ID, which is available in the Registry Report that is generated after you complete the Essbase Server configuration process.

Linux

1. In a console, change directory to `EPM_ORACLE_INSTANCE/epmsystem1/bin`.
2. Run these commands to update registry:

   ```
   epmsys_registry.sh updateproperty "#<Object ID>/@EnableSecureMode" true
   ```

   ```
   epmsys_registry.sh updateproperty "#<Object ID>/@EnableClearMode" false
   ```

   Be sure to replace `<Object ID>` with the Essbase Server component ID, which is available in the Registry Report that is generated after you complete the Essbase Server configuration process.

Updating Essbase SSL Settings

You customize the SSL settings for Essbase Server and clients by specifying values for the following in `essbase.cfg`.

- 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

> **Note:**
>
> In essbase.cfg, be sure to add any missing required parameters, specifically, EnableSecureMode, AgentSecurePort, and set their values.

To update essbase.cfg:

1. Copy Oracle wallet with certificates for Essbase Server to *EPM_ORACLE_INSTANCE*/EssbaseServer/essbaseserver1/bin/wallet.
   This is the only Oracle Wallet location acceptable to the Essbase Server.
2. Using a text editor, open *EPM_ORACLE_INSTANCE*/EssbaseServer/essbaseserver1/bin/essbase.cfg.
3. Enter settings as needed. Default Essbase settings are implied. If you need to change the default behavior, add the settings for the custom behavior in essbase.cfg. For example, EnableClearMode is enforced by default, by which Essbase Server is enabled to communicate over nonencrypted channel. To turn off Essbase Server's ability to communicate over unencrypted channel, you should specify EnableClearMode FALSE in essbase.cfg. See the following table.

Table 2-2 Essbase SSL Settings

Setting	Description [1]
EnableClearMode[2]	Enables unencrypted communication between Essbase applications and Essbase Agent. If this property is set to FALSE, Essbase does not handle non-SSL requests. **Default:** EnableClearMode TRUE **Example:** EnableClearMode FALSE
EnableSecureMode	Enables SSL encrypted communication between Essbase clients and Essbase Agent. This property must be set to TRUE to support SSL. **Default:** FALSE **Example:** EnableSecureMode TRUE
SSLCipherSuites	A list of cipher suites, in order of preference, to use for SSL communication. 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. **Default:** SSL_RSA_WITH_RC4_128_MD5 **Example:** SSLCipherSuites SSL_RSA_WITH_AES_256_CBC_SHA256,SSL_RSA_WITH_AES_256_GCM_SHA384
APSRESOLVER	URL of Oracle Hyperion Provider Services. If you are using several Provider Services servers, separate each URL using a semicolon. **Example:** APSRESOLVER https://exampleAPShost1:*PORT*/aps;https://exampleAPShost2:*PORT*/aps

Table 2-2 (Cont.) Essbase SSL Settings

Setting	Description [1]
`AgentSecurePort`	The secure port at which the agent listens. **Default:** `6423` **Example:** `AgentSecurePort 16001`
`WalletPath`	Location of the Oracle Wallet (fewer than 1,024 characters) that stores the root CA certificate and signed certificate. **Default:** `*ARBORPATH*/bin/wallet` **Example:** `WalletPath/usr/local/wallet`
`ClientPreferredMode`[3]	The mode (Secure or Clear) for the client session. If this property is set to Secure, SSL mode is used for all sessions. If this property is set to Clear, transport is chosen based on whether the client login request contains the secure transport keyword. See Establishing a Per-Session SSL Connection. **Default:** `CLEAR` **Example:** `ClientPreferredMode SECURE`

[1] The default value is enforced if these properties are not available in `essbase.cfg`.

[2] Essbase becomes nonoperational if `EnableClearMode` and `EnableSecureMode` are both set to `FALSE`.

[3] Clients use this setting to determine whether they should establish a secure or nonsecure connection with Essbase.

4. Save and close `essbase.cfg`.

Updating Distributed Essbase Nodes for SSL

> **Note:**
>
> This section applies only to distributed deployment of Essbase

Ensure that the Wallet folder (for example, `WalletPath/usr/local/wallet`) containing the root CA certificate and signed certificate is in the required location on each distributed node.

1. Copy the Wallet folder to these locations in each distributed node:
   - *`EPM_ORACLE_HOME`*`/common/EssbaseRTC/11.1.2.0/bin`
   - *`EPM_ORACLE_HOME`*`/common/EssbaseRTC-64/11.1.2.0/bin`
2. Copy the Wallet folder to these locations, if present, in each distributed node:
   - *`EPM_ORACLE_HOME`*`/products/Essbase/EssbaseServer/bin`
   - *`EPM_ORACLE_HOME`*`/products/Essbase/EssbaseServer-32/bin`
   - *`EPM_ORACLE_INSTANCE`*`/EssbaseServer/essbaseserver1/bin`
3. Copy *`EPM_ORACLE_INSTANCE`*`/EssbaseServer/essbaseserver1/bin/essbase.cfg` to these locations on each distributed node:
   - *`EPM_ORACLE_HOME`*`/common/EssbaseRTC/11.1.2.0/bin`
   - *`EPM_ORACLE_HOME`*`/common/EssbaseRTC-64/11.1.2.0/bin`

4. Copy *EPM_ORACLE_INSTANCE*/EssbaseServer/essbaseserver1/bin/essbase.cfg to these locations, if present, on each distributed node:
   - *EPM_ORACLE_HOME*/products/Essbase/EssbaseServer/bin
   - *EPM_ORACLE_HOME*/products/Essbase/EssbaseServer-32/bin
   - *EPM_ORACLE_INSTANCE*/EssbaseServer/essbaseserver1/bin
5. Copy the Wallet folder to these Essbase client installation locations on each distributed node:
   - *EPM_ORACLE_HOME*/products/Essbase/EssbaseClient/bin
   - *EPM_ORACLE_HOME*/products/Essbase/EssbaseClient-32/bin
6. Copy *EPM_ORACLE_INSTANCE*/EssbaseServer/essbaseserver1/bin/essbase.cfg to these Essbase client installation locations on each distributed node:
   - *EPM_ORACLE_HOME*/products/Essbase/EssbaseClient/bin
   - *EPM_ORACLE_HOME*/products/Essbase/EssbaseClient-32/bin
7. Add these properties to the essbase.properties file:
   - essbase.ssleverywhere=true
   - olap.server.ssl.alwaysSecure=true
   - APSRESOLVER=*http[s]://host:httpsPort/aps*
     Be sure to replace this value with the appropriate URL.

   You must update essbase.properties file at these locations, if present, in each distributed node:
   - *EPM_ORACLE_HOME*/common/EssbaseJavaAPI/11.2.0/bin/essbase.properties
   - *EPM_ORACLE_HOME*/products/Essbase/aps/bin/essbase.properties
   - *EPM_ORACLE_INSTANCE*/aps/bin/essbase.properties
8. Copy *EPM_ORACLE_HOME*/products/Essbase/aps/bin/essbase.properties to *EPM_ORACLE_HOME*/products/Essbase/eas directory, if available, on each distributed node.
9. **For Oracle Hyperion Planning Only:** Add these three properties to the essbase.properties file:
   - essbase.ssleverywhere=true
   - olap.server.ssl.alwaysSecure=true
   - APSRESOLVER=*APS_URL*
     Replace *APS_URL* with Provider Services URL. If you are using several Provider Services servers, separate each URL using a semicolon. For example, https://exampleAPShost1:*PORT*/aps;https://exampleAPShost2:*PORT*/aps.

     You must update essbase.properties file at these locations in each distributed node:
     - *EPM_ORACLE_HOME*/products/Planning/config/essbase.properties
     - *EPM_ORACLE_HOME*/products/Planning/lib/essbase.properties
10. **For Oracle Hyperion Financial Reporting Only:** Add these three properties to the *EPM_ORACLE_HOME*/products/financialreporting/bin/EssbaseJAPI/bin/essbase.properties file:
    - essbase.ssleverywhere=true

- `olap.server.ssl.alwaysSecure=true`
- `APSRESOLVER=`*`APS_URL`*
  Replace *`APS_URL`* with Provider Services URL. If you are using several Provider Services servers, separate each URL using a semicolon. For example, `https://exampleAPShost1:`*`PORT`*`/aps;https://exampleAPShost2:`*`PORT`*`/aps`.

> **Note:**
>
> In full SSL environments, Financial Reporting requires the Essbase Cluster Name to establish a connection. Connections fail if the hostname is used to connect.

11. a. Set the environment variables:
    - **Windows:** Create a new system variable named `API_DISABLE_PEER_VERIFICATION` and set its value to `1`.
    - **Linux:** Add the directive `API_DISABLE_PEER_VERIFICATION=1` in `setCustomParamsPlanning.sh`.

    b. Add the directive `API_DISABLE_PEER_VERIFICATION=1` in *`EPM_ORACLE_INSTANCE`*`/EssbaseServer/essbaseserver1/bin/setEssbaseenv.bat` or *`EPM_ORACLE_INSTANCE`*`/EssbaseServer/essbaseserver1/bin/setEssbaseenv.sh`.

    Set environment variables:

Customizing SSL Properties of JAPI Clients

Several default properties are predefined for the Essbase components that rely on JAPI. The default properties can be overridden by including properties in `essbase.properties`.

> **Note:**
>
> Only a few of the SSL properties identified in the following table are externalized in `essbase.properties`. You should add the properties that are not externalized.

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. Update properties as needed. See the following table for description of customizable JAPI client properties.
   If a desired property is not included in `essbase.properties`, add it.

Table 2-3 Default SSL properties for JAPI Clients

Property	Description
`olap.server.ssl.alwaysSecure`	Sets the mode that clients should use against all Essbase instances. Change this property value to `true` to enforce SSL mode. **Default:** `false`

Table 2-3 (Cont.) Default SSL properties for JAPI Clients

Property	Description
`olap.server.ssl.securityHandler`	Package name for handling the protocol. You can change this value to indicate another handler. **Default:** `java.protocol.handler.pkgs`
`olap.server.ssl.securityProvider`	Oracle uses the Sun SSL protocol implementation. You can change this value to indicate another provider. **Default:** `com.sun.net.ssl.internal.www.protocol`
`olap.server.ssl.supportedCiphers`	A comma-separated list of additional ciphers to be enabled for secure communication. You must specify only ciphers that Essbase supports. **Example**: `SSL_RSA_WITH_AES_256_CBC_SHA256,SSL_RSA_WITH_AES_256_GCM_SHA384`
`olap.server.ssl.trustManagerClass`	The TrustManager class to use to validate the SSL certificate by verifying the signature and checking the 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 `com.essbase.services.olap.security.EssDefaultTrustManager`, 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 `javax.net.ssl.X509TrustManager` interface. **Example**:`com.essbase.services.olap.security.EssDefaultTrustManager`

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

Establishing a Per-Session SSL Connection

Oracle 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:

```
login admin example_password on hostA:PORT:secure
```

```
login admin example_password 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 Essbase. If the `ClientPreferredMode` setting is not set to secure, the communication occurs over a nonsecure channel.

SSL for Essbase 21c

Overview

This section explains the procedures for replacing the default certificates that are used to secure communication between an Oracle Essbase instance and components such as MaxL, Oracle Essbase Administration Services Server, Oracle Hyperion Provider Services, Oracle Hyperion Foundation Services, Oracle Hyperion Planning, Oracle Hyperion Financial Management, and Oracle Hyperion Shared Services Registry.

Note:

- Essbase Administration Services (EAS) Lite does not use the HTTP Server SSL port (for example, 443) configured using the EPM Configurator. The secure URL in the `easconsole.jnlp` file defaults to the non-SSL port (80).

  Workaround: Replace the default non-SSL port in the secure URL identified in `easconsole.jnlp`with the updated secure URL:

  Default Secure URL: `https://myserver:SECURE_PORT/easconsole/console.html`. Example, `https://myserver:80/easconsole/console.html`

  Updated Secure URL: `https://myserver:SECURE_PORT/easconsole/console.html`. Example, `https://myserver:443/easconsole/console.html`

  See My Oracle Support (MOS) article - Doc ID 1926558.1 - SSL Port Not Included In easconsole.jnlp of the EAS Web Console for more information.

- Planning's connection to Essbase might not function if SSL is set up for EPM in a Windows environment.

  Workaround: Add the following folder path to the `PATH` system environment variable on the server where Planning is installed.

  ```
  EPM_ORACLE_HOME\bin21C
  ```

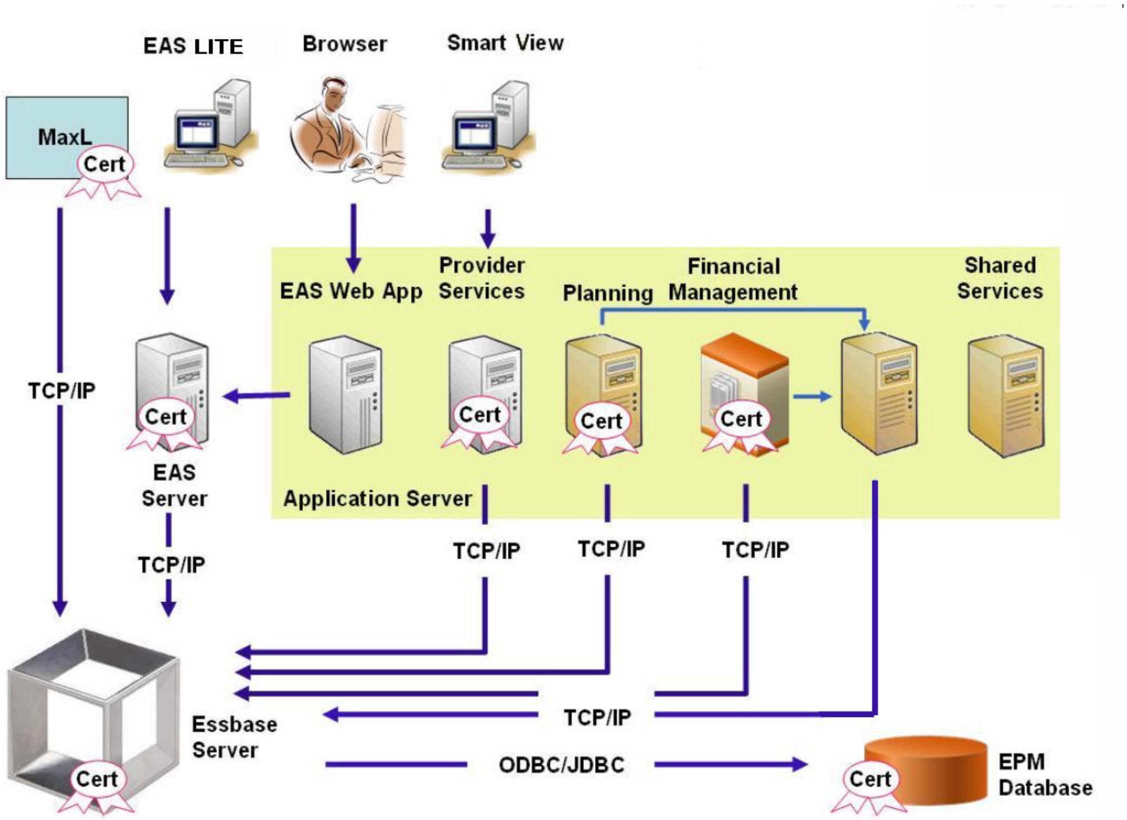
Default Deployment

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.

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 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.

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:**
>
> Essbasesupports certificates that are supported by OpenSSL.

You require the following certificates:

- A root CA certificate.
  Components that use Essbase RTC to establish a connection to Essbase require that the root CA certificate be stored in an Oracle Wallet. Components that use JAPI to establish a connection require that the root CA certificate be stored in a Java keystore. The required certificates and their locations 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 2-4 Required Certificates and Their Locations

Component[1]	Keystore	Certificate [2]
MaxL	Oracle Wallet	Root CA certificate
Administration Services Server	Oracle Wallet	Root CA certificate
Provider Services	Oracle Wallet	Root CA certificate
Oracle Enterprise Performance Management System Database	Oracle Wallet	Root CA certificate
Planning	• Oracle Wallet • Java Keystore	Root CA certificate
Financial Management	Java Keystore	Root CA certificate
Essbase (Server and Agent) [3]	• Oracle Wallet • Java Keystore	• Root CA certificate • Signed certificate for Essbase Server and Agent
Oracle Hyperion Shared Services Repository		

[1] You need only one instance of the keystore to support multiple components that use a 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 in the Java keystore.

Installing and Deploying Essbase Components

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

The EPM 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

Creating Certificate Requests and Obtaining Certificates

Generate a certificate request to obtain a certificate for the server that hosts Oracle Essbase Server and Essbase Agent. A certificate request contains encrypted information specific to your server's Common Name (CN=). 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 for the tool that you are using.

Examples using keytool:

Create a Java Keystore (JKS) and generate a private key:

```
keytool.exe -genkey -dname "cn=myserver, ou=EPM, o=Oracle, c=US"
-alias essbase_ssl -keypass password -keystore
C:\oracle\Middleware\EPMSystem11R1\ssl\EPM.JKS -storepass password
-validity 365 -keyalg RSA -keysize 2048 -sigalg SHA256withRSA -noprompt
```

Generate a certificate request:

```
keytool -certreq -alias essbase_ssl -file
C:\oracle\Middleware\EPMSystem11R1\ssl\essabse_server.csr -keypass password
-keystore C:\oracle\Middleware\EPMSystem11R1\ssl\EPM.JKS -storepass password
```

Export your private key (you will need the `openssl` utility to perform these steps):

1. `openssl.exe pkcs12 -in C:\oracle\Middleware\EPMSystem11R1\ssl\EPM.JKS -passin pass:password -legacy -nocerts -out c:\Apache24\ssl\Apache24.key -passout pass:password`
2. Sign your newly generated Certificate Request using your CA (Certifying Authority) and paste it into the following file:
   `C:\oracle\Middleware\EPMSystem11R1\ssl\essbase.cer`.

Obtaining and Installing Root CA Certificate

The root CA certificate verifies 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 .

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

Installing CA-Signed Certificates

For Installing CA-Signed Certificates, see the following links:

- Set up Weblogic TLS Connection for Essbase
- Update TLS Certificates

Update the `tls.properties` file under

```
%EPM_HOME%\essbase\bin\tls_tools.properties:
certCA=c:\\ssl\\ca.crt;c:\\ssl\\intermediate.crt;c:\\ssl\\essbase.key;c:\\ssl\
\essbase.cer;
```

Where:

```
C:\ssl\ca.crt - root CA certificate.
C:\ssl\intermediate.crt - intermediate CA certificate.
C:\ssl\essbase.key - your private key generated in the previous step.
C:\ssl\essbase.cer - your server’s signed certificate issued by your CA.
```

Execute the following to update Essbase server with the new certificates:

```
set ORACLE_HOME=c:\OracleSSL
set EPM_HOME=%ORACLE_HOME%
set WL_HOME=%ORACLE_HOME%\wlserver
set JAVA_HOME=%ORACLE_HOME%\jdk
set DOMAIN_HOME=%ORACLE_HOME%\user_projects\domains\essbase_domain
%EPM_HOME%\essbase\bin\tls_tools.properties:
%ORACLE_HOME%\\jdk\bin\java.exe -Xmx256m -jar %ORACLE_HOME%
\essbase\lib\tlsTools.jar %EPM_HOME%\essbase\bin\tls_tools.properties
```

Updating Essbase SSL Settings

You customize the SSL settings for Essbase Server and clients by specifying values for the following in `essbase.cfg`.

- 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

> **Note:**
>
> In `essbase.cfg`, be sure to add any missing required parameters, specifically, `EnableSecureMode`, `AgentSecurePort`, and set their values.

To update `essbase.cfg` located under:

```
ESSBASE_DOMAIN_HOME\config\fmwconfig\essconfig\essbase
```

1. Enter settings as needed. Default Essbase settings are implied. If you need to change the default behavior, add the settings for the custom behavior in `essbase.cfg`. For example, `EnableClearMode` is enforced by default, by which Essbase Server is enabled to communicate over nonencrypted channel. To turn off Essbase Server's ability to communicate over unencrypted channel, you should specify `EnableClearMode FALSE` in `essbase.cfg`. See the following table:

Table 2-5 Essbase SSL Settings

Setting	Description [1]
`EnableClearMode`[2]	Enables unencrypted communication between Essbase applications and Essbase Agent. If this property is set to `FALSE`, Essbase does not handle non-SSL requests. **Default:** `EnableClearMode TRUE` **Example:** `EnableClearMode FALSE`
`EnableSecureMode`	Enables SSL encrypted communication between Essbase clients and Essbase Agent. This property must be set to `TRUE` to support SSL. **Default:** `FALSE` **Example:** `EnableSecureMode TRUE`
`SSLCipherSuites`	A list of cipher suites, in order of preference, to use for SSL communication. 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. **Default:** `SSL_RSA_WITH_RC4_128_MD5` **Example:** `SSLCipherSuites SSL_RSA_WITH_AES_256_CBC_SHA256,SSL_RSA_WITH_AES_256_GCM_SHA384`
`APSRESOLVER`	URL of Oracle Hyperion Provider Services. If you are using several Provider Services servers, separate each URL using a semicolon. **Example:** `https://exampleAPShost1:`*`PORT`*`/essbase;https://exampleAPShost2:`*`PORT`*`/essbase`
`AgentSecurePort`	The secure port at which the agent listens. **Default:** `6423` **Example:** `AgentSecurePort 16001`
`WalletPath`	Location of the Oracle Wallet (fewer than 1,024 characters) that stores the root CA certificate and signed certificate. **Default:** *`ARBORPATH`*`/bin/wallet` **Example:** `WalletPath/usr/local/wallet`
`ClientPreferredMode` [3]	The mode (Secure or Clear) for the client session. If this property is set to Secure, SSL mode is used for all sessions. If this property is set to Clear, transport is chosen based on whether the client login request contains the secure transport keyword. See Establishing a Per-Session SSL Connection. **Default:** `CLEAR` **Example:** `ClientPreferredMode SECURE`

- [1] The default value is enforced if these properties are not available in `essbase.cfg`.
- [2] Essbase becomes nonoperational if `EnableClearMode` and `EnableSecureMode` are both set to `FALSE`.
- [3] Clients use this setting to determine whether they should establish a secure or nonsecure connection with Essbase.

2. Save and close `essbase.cfg`.

Updating Distributed Essbase Nodes for SSL

> **Note:**
>
> This section applies only to distributed deployment of Essbase

Ensure that the Wallet folder (for example, `WalletPath/usr/local/wallet`) containing the root CA certificate and signed certificate is in the required location on each distributed node.

1. Import all the new CA certificate using TLS tools.

   For further information, see the following links:

   - Set up Weblogic TLS Connection for Essbase
   - Update TLS Certificates

2. Go the source location: `ESSBASE_DOMAIN_HOME\config\fmwconfig\essconfig\essbase` and modify the following properties in the `essbase.properties` file:

   - `essbase.ssleverywhere=true`
   - `olap.server.ssl.alwaysSecure=true`
   - `APSRESOLVER=`*`APS_URL`*
     Replace *`APS_URL`* with the Provider Services URL. If you are using several Provider Services servers, separate each URL using a semicolon. For example:

     `https://exampleAPShost1:`*`PORT`*`/essbase;https://exampleAPShost2:`*`PORT`*`/essbase`

3. Copy `Wallet` folder, `Walletssl` folder, `essbase.cfg` file and `essbase.properties` file to the following destination paths.

Table 2-6 Destination Paths

Destination Paths	Wallet	Wallet ssl	essbase.cfg	essbase.properties
`EPM_ORACLE_HOME\common\EssbaseRTC-21C\11.1.2.0\bin`	Yes	Yes	Yes	Yes
`EPM_ORACLE_HOME\common\EssbaseJavaAPI-21C\11.1.2.0\bin`	Yes	Yes	Yes	Yes
`ESSBASE_DOMAIN_HOME\config\fmwconfig\essconfig\aps`	Yes	Yes	Yes	Yes
`ESSBASE_DOMAIN_HOME\config\fmwconfig\essconfig\essbase`	Yes	Yes	Yes	Yes
`MIDDLEWARE_HOME\essbase\products\Essbase\template_files\essbase`	Yes	Yes	Yes	Yes
`MIDDLEWARE_HOME\essbase\products\Essbase\EssbaseServer\bin`	Yes	Yes	Yes	Yes
`MIDDLEWARE_HOME\essbase\products\Essbase\aps\bin`	Yes	Yes	Yes	Yes
`MIDDLEWARE_HOME\essbase\products\Essbase\eas`	Yes	Yes	Yes	Yes

Table 2-6 (Cont.) Destination Paths

Destination Paths	Wallet	Wallet ssl	essbase.cfg	essbase.properties
`MIDDLEWARE_HOME\essbase\common\EssbaseJavaAPI\bin`	Yes	Yes	Yes	Yes
For Oracle Hyperion Financial Reporting Only `EPM_ORACLE_HOME/products/financialreporting/bin/EssbaseJAPI/bin/` **Note:** In full SSL environments, Financial Reporting requires the Essbase Cluster Name to establish a connection. Connections fail if the hostname is used to connect.	Yes	Yes	Yes	Yes

For Oracle Hyperion Planning Only
`EPM_ORACLE_HOME/products/Planning/config/`
`EPM_ORACLE_HOME/products/Planning/lib/`
	Yes	Yes	Yes	Yes

4. Set the environment variables:
   - **Windows:** Create a new system variable named `API_DISABLE_PEER_VERIFICATION` and set its value to `1`.
   - **Linux:** Add the directive `API_DISABLE_PEER_VERIFICATION=1` in `setCustomParamsPlanning.sh`.

Customizing SSL Properties of JAPI Clients

Several default properties are predefined for the Essbase components that rely on JAPI. The default properties can be overridden by including properties in `essbase.properties`.

> **Note:**
>
> Only a few of the SSL properties identified in the following table are externalized in `essbase.properties`. You should add the properties that are not externalized.

To update SSL properties of JAPI clients:

1. Using a text editor, open `EPM_ORACLE_HOME/common/EssbaseJavaAPI-21C/11.2.0/bin/essbase.properties`.
2. Update properties as needed. See the following table for description of customizable JAPI client properties.
   If a desired property is not included in `essbase.properties`, add it.

Table 2-7 Default SSL properties for JAPI Clients

Property	Description
`olap.server.ssl.alwaysSecure`	Sets the mode that clients should use against all Essbase instances. Change this property value to `true` to enforce SSL mode. **Default:** `false`

Table 2-7 (Cont.) Default SSL properties for JAPI Clients

Property	Description
`olap.server.ssl.securityHandler`	Package name for handling the protocol. You can change this value to indicate another handler. **Default:** `java.protocol.handler.pkgs`
`olap.server.ssl.securityProvider`	Oracle uses the Sun SSL protocol implementation. You can change this value to indicate another provider. **Default:** `com.sun.net.ssl.internal.www.protocol`
`olap.server.ssl.supportedCiphers`	A comma-separated list of additional ciphers to be enabled for secure communication. You must specify only ciphers that Essbase supports. **Example**: `SSL_RSA_WITH_AES_256_CBC_SHA256,SSL_RSA_WITH_AES_256_GCM_SHA384`
`olap.server.ssl.trustManagerClass`	The TrustManager class to use to validate the SSL certificate by verifying the signature and checking the 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 `com.essbase.services.olap.security.EssDefaultTrustManager`, 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 `javax.net.ssl.X509TrustManager` interface. **Example**:`com.essbase.services.olap.security.EssDefaultTrustManager`

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

Establishing a Per-Session SSL Connection

Oracle 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:

```
login admin example_password on hostA:PORT:secure
```

```
login admin example_password 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 Essbase. If the `ClientPreferredMode` setting is not set to secure, the communication occurs over a nonsecure channel.

3
Enabling SSO with Security Agents

Related Topics

- Supported SSO Methods
- Enabling OAM SSO for Essbase 21c embedded with EPM 11.2.15 and above
- Single Sign-on from Oracle Access Manager
- OracleAS Single Sign-on
- Protecting EPM System Products for SSO
- Header-based SSO with Identity Management Products
- Configuring EPM System for Header-based SSO with Oracle Identity Cloud Services
- SiteMinder SSO
- Kerberos Single Sign-on
- Configuring EPM System for SSO
- Single Sign-on Options for Smart View

Supported SSO Methods

SSO requires that the web identity management solution pass the login name of authenticated users to Oracle Enterprise Performance Management 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
- Custom Login Class
- HTTP Authorization Header
- Get Remote User from HTTP Request
- Header-based Authentication with Identity Management Products

> **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 Single Sign-on (OSSO), SiteMinder, or Oracle Access Manager as the web identity management solution, EPM System security automatically selects Custom HTTP header to pass the login name of authenticated users to EPM System components.

The login name of an EPM System product user is determined by the `Login Attribute` that is specified while configuring user directories in Oracle Hyperion Shared Services. See "Configuring OID, Active Directory, and Other LDAP-Based User Directories" in the *Oracle Enterprise Performance Management System User Security Administration 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.

Using a custom login class enables EPM System to integrate with security agents that use X509 certificate-based authentication. Using this authentication mechanism requires the implementation of standard Shared Services APIs to define the SSO interface between EPM System components 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 Security Administration Guide* for a brief description of `Login Attribute`. For sample code and implementation steps, see Implementing a Custom Login Class.

To use a custom login class (default name is `com.hyperion.css.sso.agent.X509CertificateSecurityAgentImpl`), an implementation of `com.hyperion.css.CSSSecurityAgentIF` interface must be available in the classpath. `CSSSecurityAgentIF` defines the getter method for retrieving the 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 the 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.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.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 value the of `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.

Header-based Authentication with Identity Management Products

EPM System supports any identity management product such as Oracle Identity Cloud Services, Microsoft Azure AD, Okta, that supports header-based authentication. Conceptual work flow is as follows:

- A gateway application acting as a reverse proxy protects EPM System components by restricting unauthenticated network access.
- The gateway application intercepts HTTP(S) requests to EPM System components and ensures that the identity management product authenticates users before forwarding requests to EPM System components.
- While forwarding requests to EPM System components, the gateway application propagates the authenticated user's identity to the EPM System component through HTTP header requests.

To support this authentication scenario, EPM System should be configured to work with the authenticated user's identity that is propagated through HTTP(S) header requests.

Enabling OAM SSO for Essbase 21c embedded with EPM 11.2.15 and above

For OAM (Oracle Access Manager) SSO (Single Sign-On) to function with Essbase, `OAMIdentityAsserter` must be added and configured in the WebLogic domain that Essbase uses.

Add OAMIdentityAsserter

To add `OAMIdentityAsserter` as a new Authentication Provider in the WebLogic console:

1. Log in to the WebLogic Administration Console if you are not already logged in.
2. Click **Security Realms** on the left, click **myrealm**, and then click the **Providers** tab.
3. Click **New**, and enter the following details:
   - **Name:** `OAMIdentityAsserter`
   - **Type:** `OAMIdentityAsserter`
4. Click **OK**.

Configure OAMIdentityAsserter

In the Authentication Providers table, click the provider you just created. On the **Common** tab:

1. Set the Control Flag to "Required".
2. Ensure that the Active Types selection for the SSO mechanism is appropriately chosen to include the header such as `OAM_REMOTE_USER`, which your WebGate adds after OAM authentication. This enables Identity Assertion based on the specified header.
3. Click **Save** to save the configuration.

Reorder Providers

1. Under the Authentication Providers table, click **Reorder**.
2. Select the `OAMIdentityAsserter` provider on the **Reorder Authentication Providers** page, and then use the arrows next to the list to arrange it so that it comes before `EssbaseCSSAuthenticator`.

Update Logout URL

Update or add the `LOGOUT_URL` in `$ESSBASE_DOMAIN\bin\setStartupEnv.cmd` or `$ESSBASE_DOMAIN\bin\setStartupEnv.sh` under startup group condition:

```
'if "%STARTUP_GROUP%"=="ESSBASE-MAN-SVR"'
```

Use the Fully-Qualified Domain Name (FQDN) of the OAM server and set the logout URL as below:

```
-DLOGOUT_URL=https://<oam.server.host>:<oam.server.port>/oam/server/logout?
end_url=https://<oam.server.host>:<oam.server.port>/oam/pages/logout.jsp
```

Single Sign-on from Oracle Access Manager

Oracle Enterprise Performance Management 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 Oracle Hyperion Shared Services. See "Configuring OID, Active Directory, and Other LDAP-Based User Directories" in the *Oracle Enterprise Performance Management System User Security Administration 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:**
>
> Oracle Essbase 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:

- Set up the required policy domains foEPM System components
- Configured an HTTP header to pass login attribute value to EPM System
- Protected the EPM System resources listed in Resources to Protect. Requests to access protected resources are challenged by Oracle Access Manager.
- Unprotected the EPM System resources listed in Resources to Unprotect. 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 Security Administration 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 EPM System for SSO.

   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 Data Relationship Management only:

   a. Configure Data Relationship Management for Shared Services authentication.

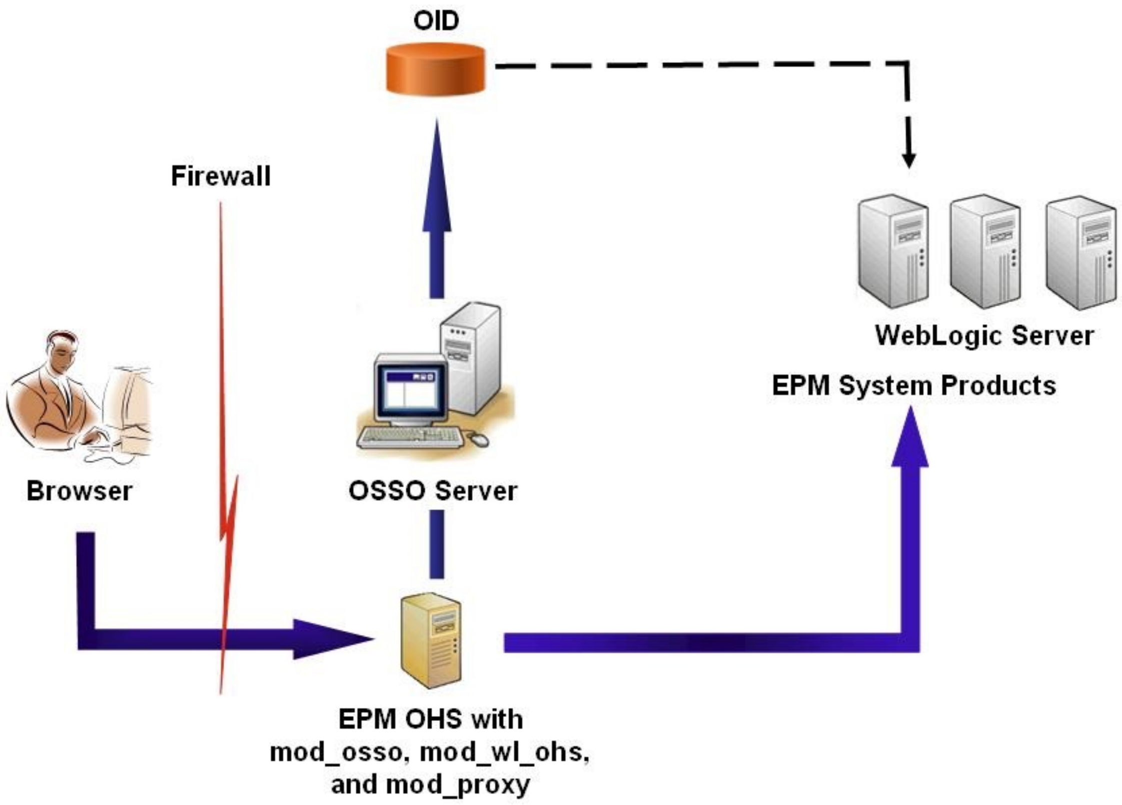
   b. Enable SSO in Data Relationship Management Console.

      See the 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 Oracle Enterprise Performance Management 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` (Oracle WebLogic 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 Oracle Hyperion 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 10*g* 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 frontend to the OSSO Server. This installation includes `mod_osso`, which 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.

2. A fully functional EPM System deployment.
   When you configure the web server for EPM System components, EPM System Configurator configures `mod_wl_ohs.conf` on the Oracle HTTP Server to proxy requests to the WebLogic Server.

Testing the Deployment

After completing the SSL deployment, verify that everything works.

To test your deployment:

1. Using a browser, access the secure Oracle Hyperion Enterprise Performance Management 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 Oracle Enterprise Performance Management System components.

Enabling OSSO for EPM System

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 Oracle Enterprise Performance Management 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 the Virtual Host.
   - `-config_file` indicates the path where `osso.conf` file is to be generated.

Optional: Defining the 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
   OssoConfigFile C:/Oracle/Middleware/user_projects/epmsystem1/httpConfig/
      ohs/config/OHS/ohs_component/osso/osso.conf
```

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.

```
   <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) creates an obfuscated `osso.conf` file 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.
2. Copy `osso.conf` into the directory (on Oracle HTTP Server that front-ends the OSSO Server) identified by the `OssoConifgFile` property defined in `mod_osso.conf` (see Creating mod_osso.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 Security Administration Guide*.
2. Enable SSO in the EPM System. Configuring EPM System for SSO

   > **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 Oracle Hyperion 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 Debugging Messages on the OSSO Server

To record debugging messages on OSSO server, modify `policy.properties`. Debugging 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 Debugging Messages for Protected Resources

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

To record debugging messages for protected resources:

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

   ```
   OraLogSeverity TRACE:32
   ```

3. Save and close `httpd.conf`.

Protecting EPM System Products for SSO

You must protect Oracle Enterprise Performance Management 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.

OAM Only: Preventing Default Headers from Being Added to Responses

By default, OAM adds two headers; Pragma: no-cache and Cache-Control: no-cache, to protected URLs. Because these headers conflict with similar caching directives added by the EPM System and web applications, browsers may not cache the content of protected URLs causing slower performance.

For detailed information on preventing these OAM headers from being added to responses, see "*Tuning OAM Agents*" in the "Oracle Access Management Performance Tuning" section of *Fusion Middleware Administrator's Guide for Oracle Access Manager with Oracle Security Token Service*.

Resources to Protect

The following table 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 Oracle Hyperion Enterprise Performance Management Workspace and Oracle Hyperion Financial Reporting you need to set only the parameters indicated in the following example:

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

Table 3-1 EPM System Resources to Protect

EPM System Product	Context to Protect
Oracle Hyperion Shared Services	• /interop • /interop/.../*
Oracle Hyperion EPM Workspace	• /workspace • /workspace/.../*
Oracle Hyperion Financial Reporting	• /hr • /hr/.../*
Oracle Hyperion Planning	• /HyperionPlanning • /HyperionPlanning/.../*
Oracle Integrated Operational Planning	• /interlace • /interlace/.../*
Oracle Hyperion Financial Management	• /hfmadf • /hfmadf/.../*
Oracle Hyperion Financial Reporting Web Studio	/frdesigner/**
Oracle Data Relationship Management	• /drm-web-client • /drm-web-client/.../*
Oracle Essbase Administration Services	• /hbrlauncher • /hbrlauncher/.../*
Oracle Hyperion Financial Data Quality Management	• /HyperionFDM • /HyperionFDM/.../*
Oracle Hyperion Calculation Manager	• /calcmgr • /calcmgr/.../*
Oracle Hyperion Provider Services	• /aps • /aps/.../*
Oracle Hyperion Profitability and Cost Management	• /profitability • /profitability/.../*
Account Reconciliation Manager	• /arm • /arm/.../*
Oracle Hyperion Financial Close Management	• /fcc • /fcc/.../*
Oracle Hyperion Financial Data Quality Management, Enterprise Edition	• /aif • /aif/.../*

Table 3-1 (Cont.) EPM System Resources to Protect

EPM System Product	Context to Protect
Oracle Hyperion Tax Governance	/tss
Tax Operations	/taxop
Oracle Hyperion Tax Provision	/taxprov
Supplemental Data Manager	• /sdm* • /sdm/** • /sdm/../** • /SDM-Datamodel-context-root/**
Oracle Essbase	• /essbase/.../* • /essbase/** • /essbase*

Resources to Unprotect

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

```
<LocationMatch /interop/framework(.*)>
   Require valid-user
   AuthType Basic
   allow from all
   satisfy any
</LocationMatch>
```

Table 3-2 EPM System Resources to Unprotect

EPM System Product	Contexts to Unprotect
Oracle Hyperion Shared Services	• /interop/framework • /interop/framework* • /interop/framework.* • /interop/framework/.../* • /interop/Audit • /interop/Audit* • /interop/Audit.* • /interop/Audit/.../* • /interop/taskflow • /interop/taskflow* • /interop/taskflow/.../* • /interop/WorkflowEngine • /interop/WorkflowEngine/* • /interop/WorkflowEngine/.../* • /interop/TaskReceiver • /framework/lcm/HSSMigration
Oracle Hyperion EPM Workspace	• /epmstatic/.../* • /workspace/bpmstatic/.../* • /workspace/static/.../* • /workspace/cache/.../*

Table 3-2 (Cont.) EPM System Resources to Unprotect

EPM System Product	Contexts to Unprotect
Oracle Hyperion Planning	• /HyperionPlanning/Smartview • /HyperionPlanning/faces/PlanningCentral • /HyperionPlanning/servlet/HspDataTransfer • /HyperionPlanning/servlet/HspLCMServlet • /HyperionPlanning/servlet/HspADMServlet/…/* • /HyperionPlanning/servlet/HspADMServlet/** • /HyperionPlanning/servlet/HspADMServlet* • /HyperionPlanning/servlet/HspAppManagerServlet/…/* • /HyperionPlanning/servlet/HspAppManagerServlet/** • /HyperionPlanning/servlet/HspAppManagerServlet*
Oracle Hyperion Financial Reporting	• /hr/common/HRLogon.jsp • /hr/services • /hr/services/* • /hr/services/.../* • /hr/modules/com/hyperion/reporting/web/reportViewer/HRStaticReport.jsp • /hr/modules/com/hyperion/reporting/web/repository/HRObjectListXML.jsp • /hr/modules/com/hyperion/reporting/web/reportViewer/HRHtmlReport.jsp • /hr/modules/com/hyperion/reporting/web/bookViewer/HRBookTOCFns.jsp • /hr/modules/com/hyperion/reporting/web/bookViewer/HRBookPdf.jsp
Oracle Data Relationship Management	/drm-migration-client
Oracle Hyperion Calculation Manager	• /calcmgr/importexport.postExport.do • /calcmgr/common.performAction.do • /calcmgr/lcm.performAction.do* • /calcmgr/lcm.performAction.do/*
Oracle Hyperion Financial Management	/oracle-epm-fm-webservices
Oracle Hyperion Financial Close Management	• /FCC-DataModel-context-root • /oracle-epm-erpi-webservices/* • /ARM-DataModel-context-root • /oracle-epm-erpi-webservices/** • /arm/batch/armbatchexecutionservlet • /ARM-DataModel-context-root

Table 3-2 (Cont.) EPM System Resources to Unprotect

EPM System Product	Contexts to Unprotect
Integrated Operational Planning	• /interlace/services/ • /interlace/services/* • /interlace/services/.* • /interlace/services/.../* • /interlace/anteros • /interlace/anteros/* • /interlace/anteros/.* • /interlace/anteros/.../* • /interlace/interlace • /interlace/interlace/* • /interlace/interlace/.* • /interlace/interlace/.../* • /interlace/WebHelp • /interlace/WebHelp/* • /interlace/WebHelp/.* • /interlace/WebHelp/.../* • /interlace/html • /interlace/html/* • /interlace/html/.* • /interlace/html/.../* • /interlace/email-book • /interlace/email-book/* • /interlace/email-book/.* • /interlace/email-book/.../*
Oracle Hyperion Profitability and Cost Management	• /profitability/cesagent • /profitability/lcm • /profitability/control • /profitability/ApplicationListener • /profitability/HPMApplicationListener
Oracle Essbase	• /essbase/agent • /essbase/agent* • /essbase/agent/.../* • /essbase/Essbase • /essbase/Essbase* • /essbase/Essbase/.../* • /essbase/jet/logout.html • /essbase/jet/.+\.(js|css|gif|jpe?g|png)$
Oracle Hyperion Financial Data Quality Management, Enterprise Edition	• /aif/services/FDMRuleService • /aif/services/RuleService • /aif/LCMServlet

Resources to Exclude

The following table lists the contexts that must be excluded.

Table 3-3 EPM System Resources to Exclude

EPM System Product	Contexts to Exclude
Oracle Essbase Administration Services	• /eas • /eas* • /eas/** • /eas/.../* • /easconsole • /easconsole* • /easconsole/** • /easconsole/.../* • /easdocs • /easdocs* • /easdocs/** • /easdocs/.../*

Header-based SSO with Identity Management Products

Prerequisites

- A fully configured Oracle Enterprise Performance Management System. The directory server of the identity management product must be configured in EPM System as a user directory to authorize users.
- A fully configured identity management product (Microsoft Azure AD, Okta, and so on) that supports header-based authentication.

The following generic processes are involved in configuring EPM System for header-based SSO with a compatible identity management product. Because the specific steps involved depend on the product that you are using, consult your identity management product manuals for detailed steps.

For detailed steps on configuring header-based authentication with Oracle Identity Cloud Services, see Configuring EPM System for Header-based SSO with Oracle Identity Cloud Services.

1. Register EPM System as an enterprise application in the identity management product.
   This step allows the identity management administrator to configure authentication on the enterprise application including supported features like multi-factor authentication.
   Use the Fully-Qualified Domain Name (FQDN) of the gateway appended with `workspace/index.jsp` (example, `https://gateway.server.example.com:443/workspace/index.jsp`) as the enterprise application URL for EPM System.

   Configure the EPM System enterprise application to propagate an HTTP header.
   You can choose any unreserved header name as the name of the HTTP header. The value of the header should be the property that uniquely identifies EPM System users.
2. Install, configure and register an application gateway to ensure that the enterprise application forwards only authenticated requests to EPM System.
   Use the following configuration settings:
   - FQDN of the gateway server (example, `gateway.server.example.com:443`) as the access point.
   - FQDN of EPM System (example, `epm.server.example.com:443`) as the resource to which authenticated HTTP(S) requests should be forwarded.

3. Enable SSO in EPM System to honor HTTP(S) headers propagated by the application gateway. For detailed information, see Setting Security Options.
   To enable SSO:
   a. Access Oracle Hyperion Shared Services Console as a System Administrator. See Launching Shared Services Console.
   b. Select **Administration**, and then **Configure User Directories**.
   c. Click **Security Options**.
   d. In the **Single Sign-On Configuration** section:
      i. Select the **Enable SSO** check box.
      ii. From **SSO Provider or Security Agent** drop-down list, select **Other**.
      iii. From **SSO Mechanism** drop-down list, select **Custom HTTP Header** and then specify the name of the header that the security agent passes to EPM System.
   e. Click **OK**.
4. Update Oracle Hyperion Enterprise Performance Management Workspace Post Logoff URL setting to that of the web page that you want users to see when they log out of EPM System.
   To update Post Logoff URL setting in EPM Workspace:
   a. Access EPM Workspace as a System Administrator. See Accessing EPM Workspace.
   b. Select **Navigate**, then **Workspace Settings**, and then **Server Settings**.
   c. In **Workspace Server Settings**, change **POST Logoff URL** to the URL of the web page that you want users to see when they log out of EPM System.
   d. Click **OK**.
5. Restart Oracle Hyperion Foundation Services and all EPM System managed servers.

Configuring EPM System for Header-based SSO with Oracle Identity Cloud Services

In this scenario, Oracle Identity Cloud Services authenticates Oracle Enterprise Performance Management System users and propagates the required HTTP headers to enable SSO.

This section discusses the steps involved in setting up and configuring EPM System to support SSO with Oracle Identity Cloud Services. You can extrapolate these steps to support header-based authentication of EPM System with any identity management system (for example, Azure AD) or Infrastructure as a Service (IaaS) provider that supports header-based authentication.

Conceptual work flow is as follows:

- A gateway application acting as a reverse proxy protects EPM System components by restricting unauthenticated network access.
- The gateway application intercepts HTTP(S) requests to EPM System components and ensures that the identity management product authenticates users before forwarding requests to EPM System components.
- While forwarding requests to EPM System components, the gateway application propagates the authenticated user's identity to the EPM System component through HTTP header requests.

Prerequisites and Sample URLs

To establish header-based SSO with Oracle Identity Cloud Services:

- A fully configured Oracle Enterprise Performance Management System.
- A host or container with a fully configured Oracle App Gateway, which acts as a reverse proxy to protect EPM System by restricting unauthorized access.
  Oracle App Gateway should be configured to intercept HTTP requests to EPM System components and ensure that users are authenticated by Oracle Identity Cloud Services before forwarding requests toEPM System. While forwarding requests to EPM System components, Oracle App Gateway should propagate the authenticated user's identity through HTTP Header requests.
- Domain Administrator access to Oracle Identity Cloud Services.

The following sample URLs are used in this discussion:

- Fully Qualified Domain Name (FQDN) base URL of Oracle Identity Cloud Services server (identity provider):
  `https://identity.server.example.com:443/`
- FQDN of Oracle App Gateway server (that hosts the gateway application):
  `https://gateway.server.example.com:443/`
- Enterprise application URL for EPM System. This is the FQDN of Oracle App Gateway server appended with `workspace/index.jsp`:
  `https://gateway.server.example.com:443/workspace/index.jsp`

> **Note:**
>
> Oracle Identity Cloud Services and Oracle App Gateway are configured with HTTPS support. HTTPS support for EPM System is optional.
> This discussion assumes that EPM System has been configured with HTTPS support.

Enabling Header-Based Authentication for EPM System

Enabling header-based authentication for Oracle Enterprise Performance Management System involves the following steps:

- Adding EPM System Application and Gateway to Oracle Identity Cloud Services
- Configuring the App Gateway
- Configuring User Directory for Authorization
- Enabling SSO in EPM System
- Updating EPM Workspace Settings

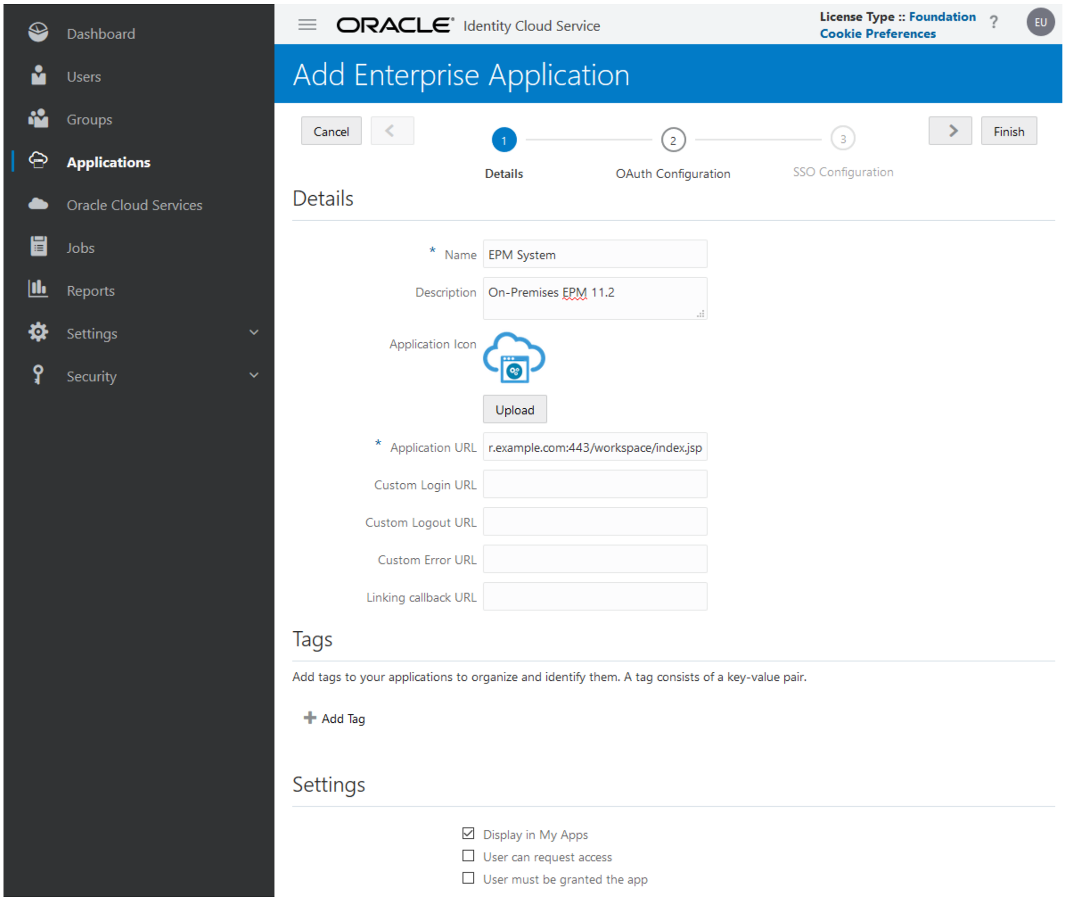
Adding EPM System Application and Gateway to Oracle Identity Cloud Services

To set up header-based authentication, you need to create Oracle Enterprise Performance Management System as an Enterprise Application.

Add EPM System as an Enterprise Application in Oracle Cloud Identity Console

To add EPM System as an enterprise application:

1. Access Oracle Cloud Identity Console as a Domain Administrator.
   a. Using a browser, go to `https://www.oracle.com/cloud/sign-in.html`.
   b. Enter your Oracle Fusion Cloud EPM Account Name.
   c. In Oracle Fusion Cloud EPM Account Sign In page, enter your user name and password, and then click **Sign In**.
   d. In the **Navigation Drawer**, click **Users**, and then **Identity (Primary)**.
   e. Click **Identity Console**.
2. Add EPM System as an Enterprise Application.
   a. In the Navigation drawer, click **Applications**.
   b. Click **Add**, and then **Enterprise Application**.



Add Enterprise Application

Details

* Name: EPM System

Description: On-Premises EPM 11.2

Application Icon

* Application URL: r.example.com:443/workspace/index.jsp

Custom Login URL:

Custom Logout URL:

Custom Error URL:

Linking callback URL:

Tags

Add tags to your applications to organize and identify them. A tag consists of a key-value pair.

Settings

☑ Display in My Apps
☐ User can request access
☐ User must be granted the app

3. Add application details:
   a. In **Name**, enter a unique name to identify EPM System enterprise application.
   b. Enter an optional description.
   c. Optionally, upload an application icon for EPM System. Click **Upload** to select and upload the icon.

d. In **Application URL**, enter the launch URL to which the gateway should redirect users. This URL is the FQDN of the Oracle App Gateway appended with `workspace/index.jsp`, which is the EPM System application context.

e. Under **Settings**, select **Display in My Apps** to display the EPM System enterprise application on the **SSO Configuration** tab of the **My Apps** page in Oracle Cloud Identity Console.

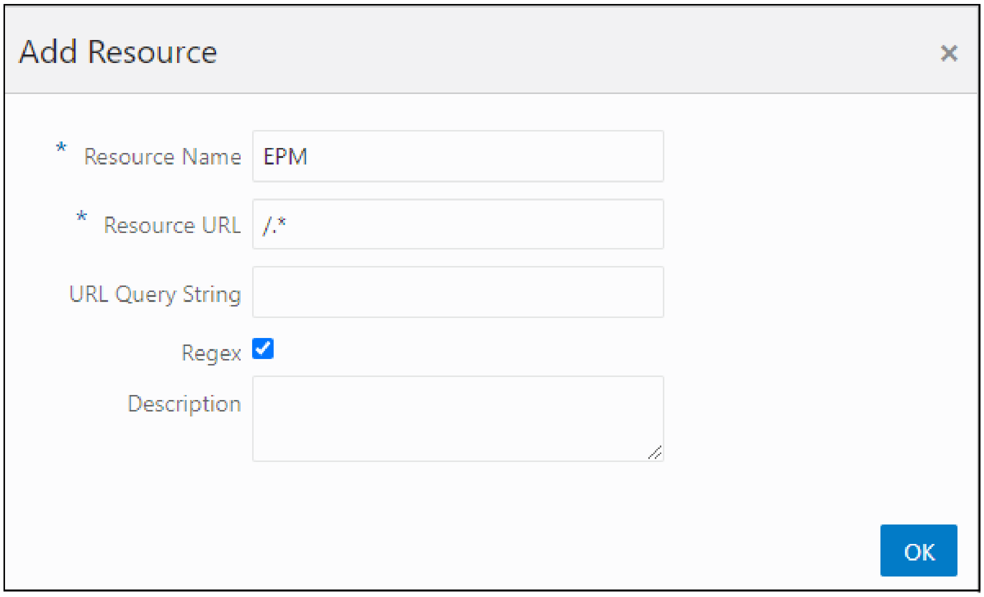
f. Click **Next**.

4. Specify SSO Configuration details.

a. Click **SSO Configuration**.

b. Add a resource for the enterprise application.
In **SSO Configuration**, expand **Resources**.

i. Click **Add**.



Add Resource

* Resource Name: EPM

* Resource URL: /.*

URL Query String:

Regex: ☑

Description:

ii. Specify a unique resource name.

iii. In **Resource URL**, enter `/.*`.

iv. Select the **Regex** check box.

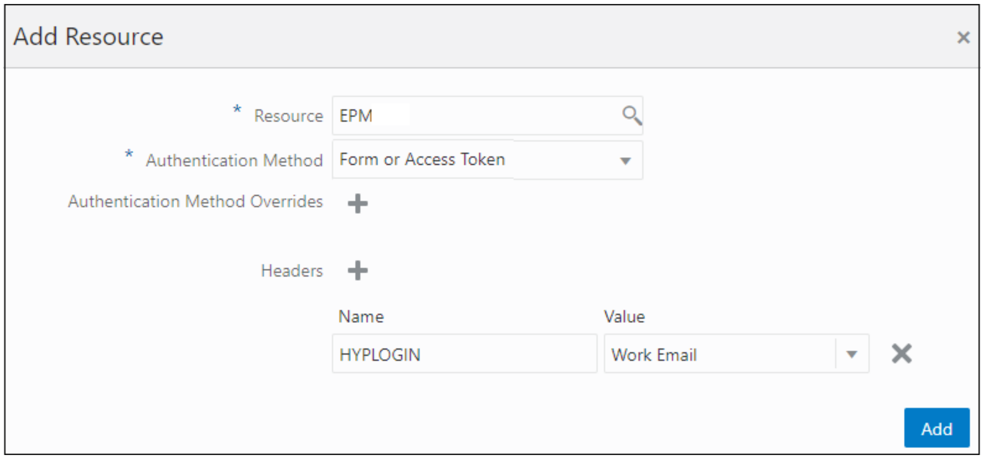
v. Click **OK**.

vi. In **SSO Configuration**, expand **Resources**.

c. Add authentication policy.
In **SSO Configuration**, expand **Authentication Policy**.

i. Select **Allow CORS** and **Require Secure Cookies** check boxes.

ii. Click **Add** under **Managed Resources** and define **Form or Access Token** as the authentication method for the SSO resource.



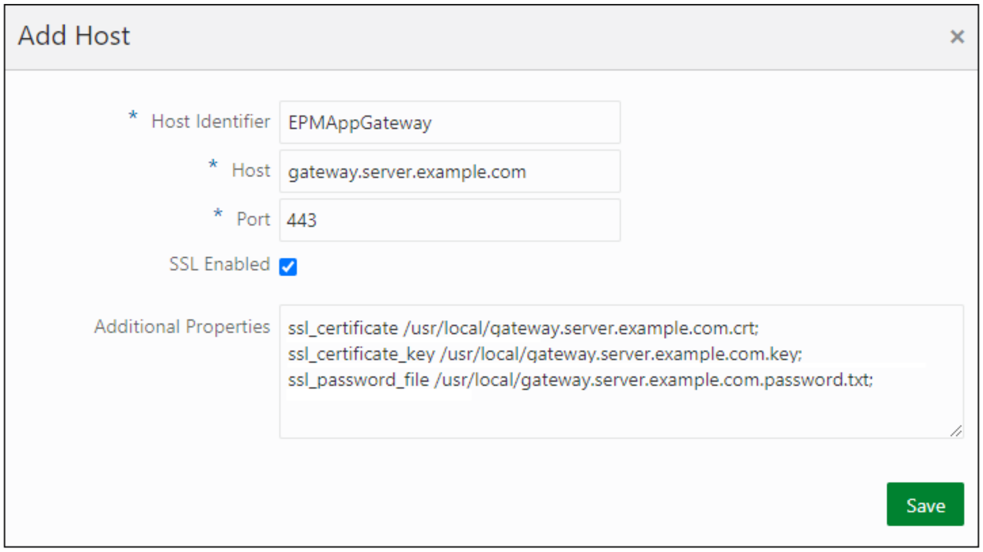
* Resource: EPM

* Authentication Method: Form or Access Token

Name	Value
HYPLOGIN	Work Email

- iii. In **Resource**, select the SSO resource that you added in the preceding step.
- iv. Expand **Headers**.
- v. Enter the name of the HTTP header that will be propagated to EPM System. Default authentication header name is `HYPLOGIN`. You can use any name of your choice.
- vi. In **Value**, select the property that uniquely identifies EPM System users. The value of this field should match the user's identity in EPM System. For example, if the user identity in EPM System is the email id, then select Work Email as the value.
- vii. Click **Save**.

5. Click **Finish** to create the enterprise application.
6. Click **Activate** to enable the application.
7. Register an App Gateway and set up the host and app for EPM System.
   - a. In the **Navigation Drawer**, click **Security** and then **App Gateways**.
   - b. Click **Add**.
   - c. In **Details**, enter a unique name for the gateway and an optional description.
   - d. Click **Next** to open the Hosts screen.
   - e. Add an App Gateway host for EPM System.
     - i. In the Hosts screen, click **Add**.


Add Host

* Host Identifier: EPMAppGateway

* Host: gateway.server.example.com

* Port: 443

SSL Enabled: ☑

Additional Properties:

```
ssl_certificate /usr/local/gateway.server.example.com.crt;
ssl_certificate_key /usr/local/gateway.server.example.com.key;
ssl_password_file /usr/local/gateway.server.example.com.password.txt;
```

ii. In **Host Identifier**, enter `EPMAppGateway`.

iii. In **Host**, enter the fully-qualified domain name of the computer that hosts the App Gateway server, for example, `gateway.server.example.com`.

iv. In **Port**, enter the port at which the App Gateway server responds to HTTPS requests.

v. Select the **SSL Enabled** check box.

vi. In **Additional Properties**, enter the following:

- SSL certificate location
- SSL certificate key
- SSL password file (if needed)

For detailed information, see "Register an App Gateway" within "Setup an App Gateway" in *Administering Oracle Identity Cloud Service*.

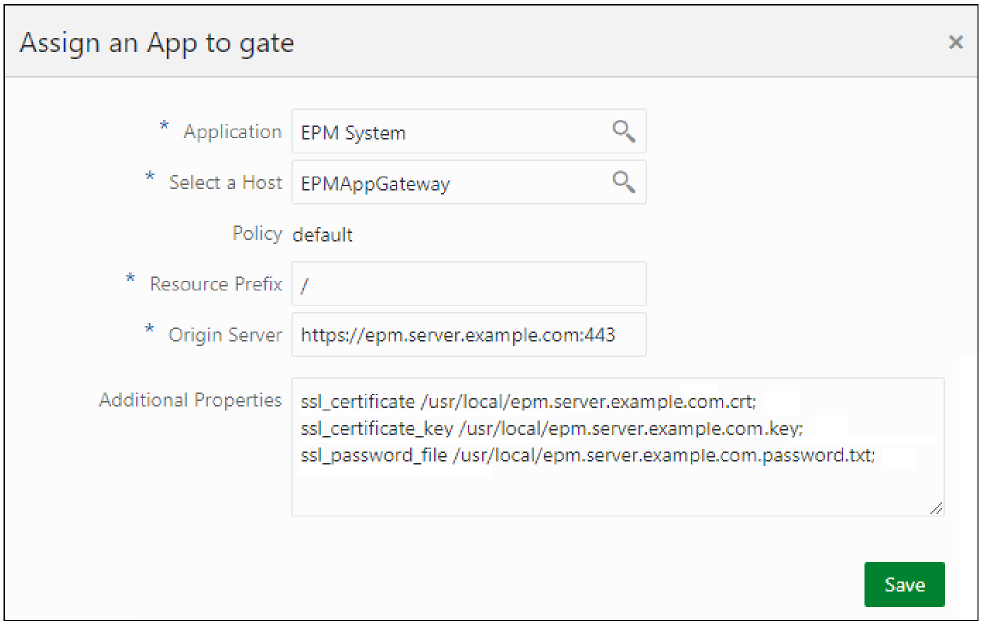
vii. Click **Save**.

viii. Click **Next** to open the Apps screen.

f. Add the EPM System enterprise application to the App Gateway.

i. On **Apps**, click **Add**.

ii. In **Application**, select the EPM System enterprise application that you previously added to Oracle Cloud Identity Console.



Assign an App to gate

* Application: EPM System

* Select a Host: EPMAppGateway

Policy: default

* Resource Prefix: /

* Origin Server: https://epm.server.example.com:443

Additional Properties:

```
ssl_certificate /usr/local/epm.server.example.com.crt;
ssl_certificate_key /usr/local/epm.server.example.com.key;
ssl_password_file /usr/local/epm.server.example.com.password.txt;
```

iii. In **Select a Host**, select `EPMAppGateway` (the EPM System host that you added to the App Gateway).

iv. In **Resource Prefix**, enter `/` to forward all requests to the EPM System host.

v. In **Origin Server**, enter the fully-qualified domain name of the computer that hosts Oracle Hyperion Enterprise Performance Management Workspace and the port number that EPM Workspace uses.

vi. Click **Save**.

8. Record the Client ID and Client Secret of the App Gateway. These values are required to set up the App Gateway.

   a. In the **Navigation Drawer**, click **Security** and then **App Gateways**.

   b. Click the name of the Gateway that you added for EPM System enterprise application.

   c. Copy the Client ID (an alpha-numeric string) to a text editor.

   d. Click **Show Secret** to display the client secret code.

   e. Copy the Client Secret (an alpha-numeric string) to the text editor.

   f. Save the text file.

> **Note:**
>
> The App Gateway server must be restarted each time a configuration update is made to the Oracle Identity Cloud Services. To start and stop the App Gateway server, see Start and Stop App Gateway.

Configuring the App Gateway

For detailed information, see "Set Up an App Gateway" in *Administering Oracle Identity Cloud Service*.

You require the Client ID and Client Secret that you recorded in the preceding section to configure the App Gateway Server.

Configuring User Directory for Authorization

Some identity management products, for example, Oracle Identity Cloud Services and Microsoft Azure, cannot be directly configured as user directories in Oracle Enterprise Performance Management System. You may configure such products with Oracle Unified Directory or Oracle Virtual Directory and then configure the latter as a user directory in EPM System. For detailed steps on configuring user directories, see Configuring User Directories.

Enabling SSO in EPM System

You configure Security Options in Oracle Enterprise Performance Management System to enable SSO. For detailed instructions, see Setting Security Options.

To enable SSO:

1. Access Oracle Hyperion Shared Services Console as a System Administrator. See Launching Shared Services Console.
2. Select **Administration**, and then **Configure User Directories**.
3. Click **Security Options**.
4. In the **Single Sign-On Configuration** section:
   a. Select the **Enable SSO** check box.
   b. From **SSO Provider or Security Agent** drop-down list, select **Other**.
   c. From **SSO Mechanism** drop-down list, select **Custom HTTP Header** and then specify the name of the header that the security agent passes to EPM System (`HYPLOGIN` or the custom name that you specified while adding resource for the enterprise application in Oracle Cloud Identity Console).
5. Click **OK**.

> **Note:**
>
> Ensure that you restart all the EPM System Services following any SSO configuration changes.

Updating EPM Workspace Settings

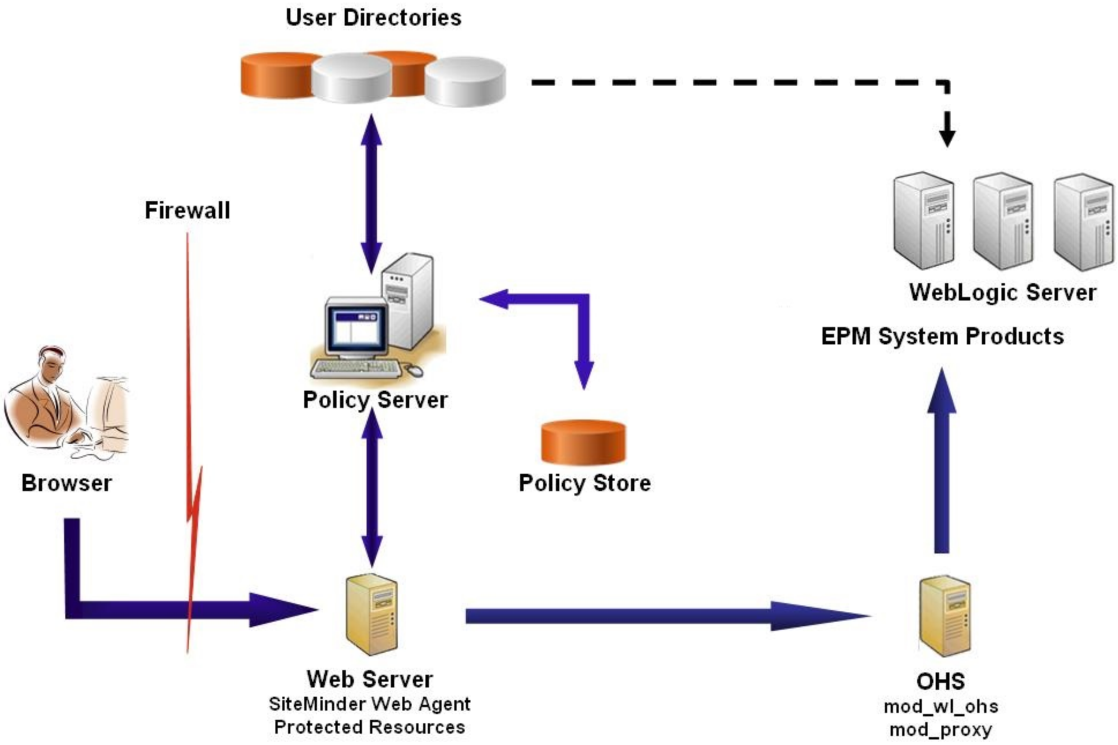
1. Access Oracle Hyperion Enterprise Performance Management Workspace as a System Administrator. See Accessing EPM Workspace.
2. Select **Navigate**, then **Workspace Settings**, and then **Server Settings**.
3. In **Workspace Server Settings**, change **POST Logoff URL** to that of the web page that you want users to see when they log out of Oracle Enterprise Performance Management System.
4. Click **OK**.
5. Restart Oracle Hyperion Foundation Services and all EPM System components.

SiteMinder SSO

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 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 Oracle Enterprise Performance Management System resource. They use a URL that connects them to the web server that front-ends the SiteMinder policy server; for example, `http://`*`WebAgent_Web_Server_Name`*`:`*`WebAgent_Web_ServerPort`*`/interop/index.jsp`.
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 Oracle WebLogic 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 are defined
   - SiteMinder Web Agent installed on the web server that front-ends the SiteMinder Policy Server
2. A fully functional EPM System deployment.
   When you configure the web server for EPM System components, EPM System Configurator configures `mod_wl_ohs.conf` to proxy requests to the WebLogic Server.

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 Oracle Hyperion 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.
2. Set the value of `enableWebAgent` property to `Yes`.

   ```
   enableWebAgent="YES"
   ```
3. Save and close the web agent configuration file.

Example 3-1 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
- Within the realm that was created for protected EPM System resources, create realms for unprotected resources. See Resources to Unprotect
- 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 Security Administration 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

Example 3-2 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.

Example 3-3 Enabling SiteMinder in EPM System

Integration with SiteMinder requires that you enable SiteMinder authentication for EPM System products. See Configuring EPM System for SSO.

Kerberos Single Sign-on

Overview

Oracle Enterprise Performance Management 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 in which each Kerberos client trusts the identities of other Kerberos clients (users, network services, and so on) to be valid.

The following happens when a user accesses an EPM System product:

1. From a Windows computer, the user logs in to a Windows domain, which is also a Kerberos realm.
2. 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.
3. 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.
4. 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 other than Oracle Smart View for Office.
- Smart View supports Kerberos integration for Oracle Essbase, Oracle Hyperion Planning, and Oracle Hyperion Financial Management providers only

Assumptions

This document, which contains application-level Kerberos configuration steps, assumes knowledge of Kerberos configuration at the system level. Before you start these procedures, confirm that the prerequisites for these tasks arecompleted.

This document assumes that you are working in a fully functional Kerberos-enabled network environment in which Windows client machines are configured for Kerberos authentication.

- The corporate Active Directory is configured for Kerberos authentication. See Microsoft Windows Server documentation.
- Browsers used to access EPM System products are configured to negotiate using Kerberos tickets.
- Time synchronization with no more than a five-minute skew between KDC and client machines. See "Authentication Errors are Caused by Unsynchronized Clocks" at http://technet.microsoft.com/en-us/library/cc780011(WS.10).aspx.

Kerberos SSO with WebLogic Server

Oracle 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:**
>
> Users must use a browser that supports the SPNEGO (for example, Internet Explorer or Firefox) to access the EPM System products running on WebLogic Server.

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.

WebLogic Server Procedures to Support Kerberos Authentication

An administrator should complete these tasks to support Kerberos authentication:

- Create the WebLogic domain for EPM System. See Creating the WebLogic Domain for EPM System.
- Create an authentication provider. See Creating an LDAP Authentication Provider in WebLogic Server.
- Create a Negotiate Identity Asserter. See Creating a Negotiate Identity Asserter.
- Create a Kerberos identification. See Creating Kerberos Identification for WebLogic Server.
- Update the JVM options for Kerberos. See Updating JVM Options for Kerberos.
- Configure authorization policies. See Configuring Authorization Policies.
- Deploy and use SSODiag to verify that WebLogic Server is ready to support Kerberos SSO for EPM System. See Using SSODiag to Test the Kerberos Environment.

Creating the WebLogic Domain for EPM System

Generally, EPM System components are deployed into `EPMSystem` WebLogic domain (the default location is *`MIDDLEWARE_HOME`*`/user_projects/domains/EPMSystem`).

To configure the EPM System WebLogic domain for Kerberos authentication:

1. Install EPM System components.
2. Deploy Oracle Hyperion Foundation Services only.
   Foundation Services deployment creates the default EPM System WebLogic domain.
3. Log in to the Oracle Hyperion Shared Services Console to verify that Foundation Services deployment was successful. See Launching Shared Services Console.

Creating an LDAP Authentication Provider in WebLogic Server

A WebLogic Server administrator creates the LDAP Authentication provider, which stores user and group information in an external LDAP server. LDAP v2- or v3-compliant LDAP servers work with WebLogic Server. See these references:

- Configuring LDAP Authentication Providers in *Oracle Fusion Middleware Securing Oracle WebLogic Server* guide.
- Configure Authentication and Identity Assertion Providers in the *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Online Help*.

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.

- Configuring a Negotiate Identity Assertion Provider in *Oracle Fusion Middleware Securing Oracle WebLogic Server* guide.
- Configure Authentication and Identity Assertion Providers in the *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Online Help*.

While creating the Negotiate Identity Assertion provider, set the JAAS Control Flag option to `SUFFICIENT` 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

On the Active Directory domain controller machine, create user objects that represent WebLogic Server and EPM System web server, and map them to service principal names (SPN) that represent your WebLogic Server and web server in the Kerberos realm. Clients cannot locate a service that does not have an SPN. You store SPNs in keytab files that are copied to the WebLogic Server domain to be used in the login process.

See Creating identification for WebLogic Server in the *Oracle Fusion Middleware Securing Oracle WebLogic Server* guide for detailed procedures.

To create Kerberos identification for WebLogic Server:

1. On the Active Directory domain controller machine, create a user account; for example, `epmHost`, for the computer that hosts the WebLogic Server domain.

   > **Note:**
   >
   > Create the identification as a user object, not as a machine.
   > Use the simple name of the computer; for example, use `epmHost` if the host is named `epmHost.example.com`.
   >
   > Record the password you use while creating the user object. You will need it to create SPNs.
   >
   > Do not select any password options, especially the `User must change password at next logon` option.

2. Modify the user object to comply with the Kerberos protocol. The account must require Kerberos pre-authentication.
   - On the **Account** tab, select an encryption to use.
   - Ensure that no other account option (especially `Do not require Kerberos pre-authentication`) is selected.
   - Because setting the encryption type may have corrupted the object's password, reset the password to the password that you set while creating the object.

3. On the computer that hosts the Active Directory domain Controller, open a command prompt window and navigate to the directory where Active Directory support tools are installed.
4. Create and configure the required SPNs.
   a. Use a command similar to the following to verify that the SPNs are associated with the user object (`epmHost`) that you created in Step 1 of this procedure.

      ```
      setspn -L epmHost
      ```

   b. Using a command such as the following, configure the SPN for WebLogic Server in Active Directory Domain Services (AD DS) and generate a keytab file that contains the shared secret key.

      ```
      ktpass -princ HTTP/epmHost.example.com@EXAMPLE.COM -pass password -
      mapuser epmHost -out c:\epmHost.keytab
      ```

5. Create a keytab file on the computer that hosts the WebLogic Server.
   a. Open a command prompt.
   b. Navigate to *`MIDDLEWARE_HOME`*`/jdk/bin`.
   c. Execute a command such as the following:

      ```
      ktab -k keytab_filename -a epmHost@example.com
      ```

   d. When prompted for a password, enter the password that you set while creating the user in step 1 of this procedure.
6. Copy the keytab file into the startup directory within the WebLogic domain; for example, into `C:\Oracle\Middleware\user_projects\domains\EPMSystem`.
7. Verify that Kerberos authentication is working correctly.

   ```
   kinit -k -t keytab-file account-name
   ```

   In this command, `account-name` indicates the Kerberos principal; for example, `HTTP/`*`epmHost.example.com@EXAMPLE.COM`*. The output from this command should be similar to the following:

   ```
   New ticket is stored in cache file C:\Documents and
   Settings\Username\krb5cc_MachineB
   ```

Updating JVM Options for Kerberos

See Using Startup Arguments for Kerberos Authentication with WebLogic Server and Creating a JAAS Login File in *Oracle Fusion Middleware Securing Oracle WebLogic Server 12c Release (12.2.1.4)*.

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

To update JVM Startup options in Windows registry:

1. Open Windows Registry Editor.

2. Select **My Computer**, then **HKEY_LOCAL_MACHINE**, then **Software**, then **Hyperion Solutions**, then **Foundationservices0**, and then **HyS9EPMServer_epmsystem1**.
3. Create the following string values:

> **Note:**
>
> The names listed in the following table are examples.

Table 3-4 JVM Startup Options for Kerberos Authentication

Name	Type	Data
JVMOption44	REG_SZ	`-Djava.security.krb5.realm=`*`Active Directory Realm Name`*
JVMOption45	REG_SZ	`-Djava.security.krb5.kdc=`*`Active Directory host name or IP address`*
JVMOption46	REG_SZ	`-Djava.security.auth.login.config=`*`location of Kerberos login configuration file`*
JVMOption47	REG_SZ	`-Djavax.security.auth.useSubjectCredsOnly=false`

4. Update the value of JVMOptionCount DWord to reflect the added JVMOptions (add 4 to the current decimal value).

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.

Using SSODiag to Test the Kerberos Environment

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

Deploying SSODiag

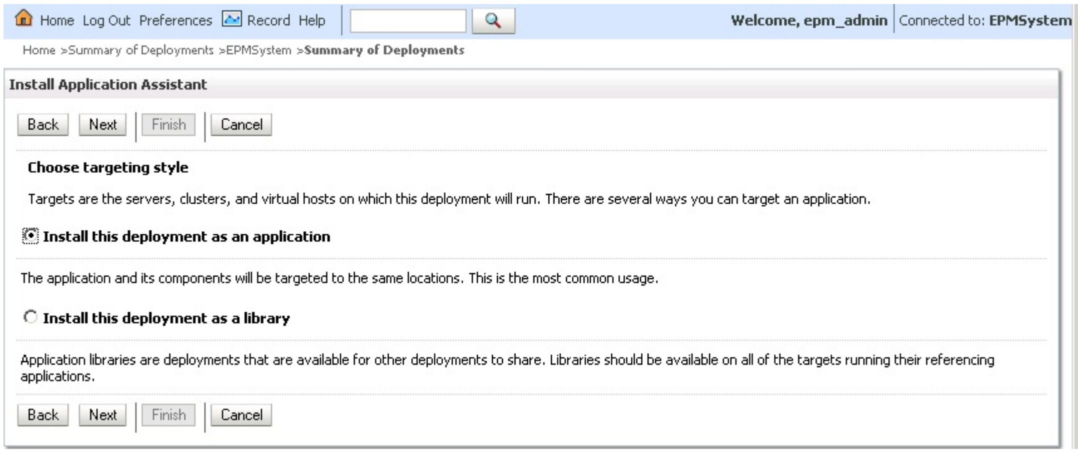
Use the WebLogic Server administrator credentials (default user name is `epm_admin`) that you specified while deploying Foundation Services to deploy SSODiag.

To deploy and configure SSOdiag:

1. Log on to the WebLogic Server Administration Console for EPM System domain.
2. In Change Center, select **Lock & Edit**
3. From **EPMSystem** in **Domain Structure**, click **Deployments**.
4. In **Summary of Deployments**, click **Install**.
5. In **Path**, select *EPM_ORACLE_HOME*`/products/Foundation/AppServer/InstallableApps/common/SSODiag.war`.

6. Click **Next**.

7. In **Choose targeting style**, ensure that **Install this deployment as an application** is selected, and then click **Next**.


Home Log Out Preferences Record Help
Welcome, epm_admin Connected to: EPMSystem
Home >Summary of Deployments >EPMSystem >Summary of Deployments
Install Application Assistant
Back Next Finish Cancel

Choose targeting style

Targets are the servers, clusters, and virtual hosts on which this deployment will run. There are several ways you can target an application.

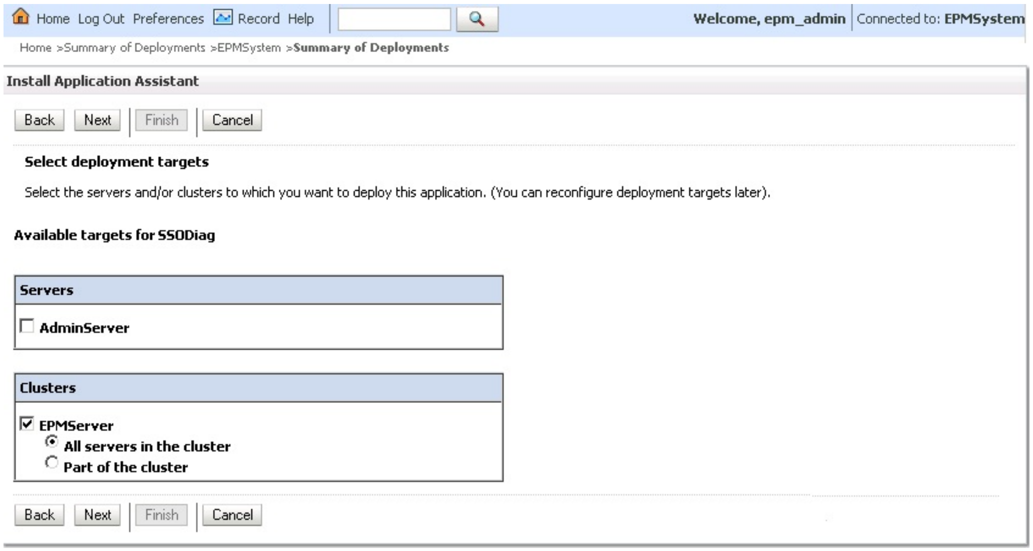
● Install this deployment as an application

The application and its components will be targeted to the same locations. This is the most common usage.

○ Install this deployment as a library

Application libraries are deployments that are available for other deployments to share. Libraries should be available on all of the targets running their referencing applications.

8. In **Select Deployment Targets**, select the following, and then click **Next**.
   - **EPMServer**
   - **All servers in the cluster**


Home Log Out Preferences Record Help
Welcome, epm_admin Connected to: EPMSystem
Home >Summary of Deployments >EPMSystem >Summary of Deployments
Install Application Assistant
Back Next Finish Cancel

Select deployment targets

Select the servers and/or clusters to which you want to deploy this application. (You can reconfigure deployment targets later).

Available targets for SSODiag

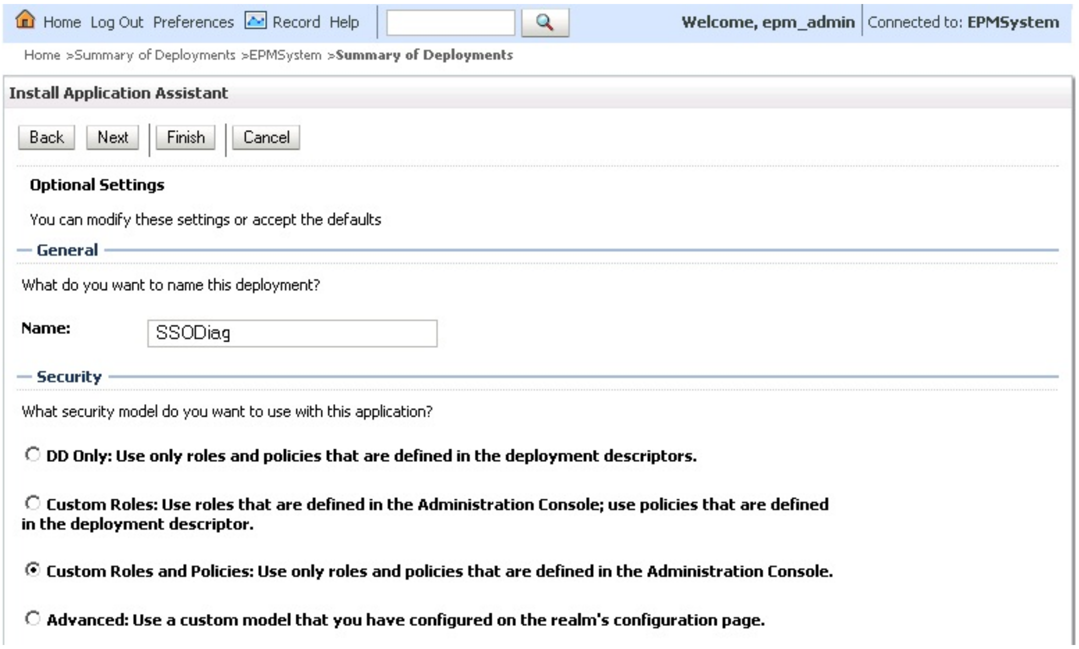
Servers

□ AdminServer

Clusters

☑ EPMServer
● All servers in the cluster
○ Part of the cluster

9. In **Optional Settings**, select **Custom Roles and Policies: Use only roles and Policies that are defined in the Administration Console** as the security model.



Install Application Assistant

Optional Settings

You can modify these settings or accept the defaults

General

What do you want to name this deployment?

Name: SSODiag

Security

What security model do you want to use with this application?

○ DD Only: Use only roles and policies that are defined in the deployment descriptors.

○ Custom Roles: Use roles that are defined in the Administration Console; use policies that are defined in the deployment descriptor.

● Custom Roles and Policies: Use only roles and policies that are defined in the Administration Console.

○ Advanced: Use a custom model that you have configured on the realm's configuration page.

10. Click **Next**.
11. On the review screen, select **No, I will review the configuration later**.
12. Click **Finish**.
13. In Change Center, select **Activate Changes**.

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.

To configure URL forwarding in Oracle HTTP Server:

1. Using a text editor, open *EPM_ORACLE_INSTANCE*`/httpConfig/ohs/config/fmwconfig/components/OHS/ohs_component/mod_wl_ohs.conf`.
2. Add a `LocationMatch` definition for SSODiag:

   ```
   <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 Oracle Hyperion 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 **URLPatterns** and then **Policies**.
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**.
   c. In **Predicate List**, select **User**.
   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**. `krbuser1` is an Active Directory or Windows desktop user.
   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.
2. Using WebLogic Server Administration Console, start the SSODiag web application to service all requests.
3. Log on to a client machine configured for Kerberos authentication using valid Active Directory credentials.
4. 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.
   ```

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, , and Other LDAP-based User Directories" in the *Oracle Enterprise Performance Management System User Security Administration Guide*.

3. Enable SSO. See Configuring OID, Active Directory, and Other LDAP-based User Directories.
   In **Security Options**, select the settings in the following table to enable Kerberos SSO.

   Table 3-5 Settings to Enable Kerberos SSO

Field	Required Setting
Enable SSO	Selected
SSO Provider or Agent	Other
SSO Mechanism	Get Remote User from HTTP Request

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 EPM System Components

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

Configuring EPM System Managed Servers for Kerberos Authentication

In Microsoft Windows environments, EPM System managed servers are run as Windows services. You must modify the startup JVM options for each WebLogic managed server. A comprehensive list of managed servers in noncompact deployment mode:

- AnalyticProviderServices0
- CalcMgr0
- ErpIntegrator0
- EssbaseAdminServer0
- FinancialReporting0
- HFMWeb0
- FoundationServices0
- HpsAlerter0
- HpsWebReports0
- Planning0
- Profitability0

If EPM System web applications are deployed in the compact deployment mode, you need to update the startup JVM options of `EPMSystem0` managed server only. If you have multiple compact managed servers, you must update the startup JVM options for all managed servers.

See Using Startup Arguments for Kerberos Authentication with WebLogic Server in the *Oracle Fusion Middleware Securing Oracle WebLogic Server* guide.

> **Note:**
>
> The following procedure describes how to set the startup JVM options for the FoundationServices managed server. You must perform this task for each WebLogic managed server in the deployment.

For detailed procedures to configure JVM options in WebLogic Server startup scripts, see Updating JVM Options for Kerberos.

To configure JVM options in WebLogic Server startup scripts

Configuring Authorization Policies

Configure authorization policies for Active Directory users who will access EPM System components other than Foundation Services. See Configuring Authorization Policies for information on configuring security policies from WebLogic Administration Console.

Changing Default Security Model of EPM System Components

You edit the EPM System configuration file to change the default security model. For non-compact EPM System deployments, you must change the default security model of each EPM System web application recorded in `config.xml`. A list of EPM System web applications:

- AIF
- APS
- CALC
- EAS
- FINANCIALREPORTING
- PLANNING
- PROFITABILITY
- SHAREDSERVICES
- WORKSPACE

To change the security model:

1. Using a text editor, open `*MIDDLEWARE_HOME*/user_projects/domains/EPMSystem/config/config.xml`
2. In the app-deployment definition of each EPM System component, set the value of `<security-dd-model>` to `CustomRolesAndPolicies` as shown in the following example:

```
<app-deployment>
   <name>SHAREDSERVICES#11.1.2.0</name>
   <target>EPMServer</target>
   <module-type>ear</module-type>
   <source-path>C:\Oracle\Middleware\EPMSystem11R1/products/Foundation/
AppServer/InstallableApps/common/interop.ear</source-path>
   <security-dd-model>CustomRolesAndPolicies</security-dd-model>
   <staging-mode>nostage</staging-mode>
</app-deployment>
```

3. Save and close `config.xml`.
4. Restart the WebLogic Server.

Creating URL Protection Policies for EPM System Components

Create a URL protection policy in the WebLogic Server Administrative Console to protect each EPM System component URL. See Options for Securing Web Applications and EJB Resources in the *Oracle Fusion Middleware Securing Resources Using Roles and Policies for Oracle WebLogic Server* guide for details.

To create URL protection policies:

1. In the Change Center in WebLogic Server Administration Console for EPM System domain, click **Lock & Edit**.
2. Click **Deployments**.
3. Expand an EPM System enterprise application (for example `PLANNING`) in your deployment and then click its web application (for example `HyperionPlanning`). See Changing Default Security Model of EPM System Components for a list of EPM System components.

   > **Note:**
   >
   > Some enterprise applications; for example Oracle Essbase Administration Services, comprise multiple web applications for which URL patters must be defined.

4. Create a URL Pattern scoped policy for the web application.
   - AIF
   - APS
   - CALC
   - EAS
   - FINANCIALREPORTING
   - PLANNING
   - PROFITABILITY
   - SHAREDSERVICES
   - WORKSPACE

   a. Click **Security**, then **Policies**, and then **New**.
   b. In **URL Pattern**, enter the protected and unprotected URLs for the EPM System products. See Protecting and Unprotecting EPM System Resources for more details.
   c. Click **OK**.
   d. Click the URL pattern that you created.
   e. Click **Add Conditions**.
   f. In **Predicate List**, select a policy condition, and then click **Next**.
   Oracle recommends that you use the `Group` condition, which grants this security policy to all members of a specified group.

g. Specify the arguments that pertain to the predicate you chose. For example, if you chose `Group` in the preceding step, you should complete the following steps:

h. In **Group Argument Name**, enter the name of the group that contains the users who should be allowed to access the web application. The name that you enter must exactly match an Active Directory group name.

   - Click **Add**.
   - Repeat the preceding steps to add more groups.

i. Click **Finish**.
   WebLogic Server displays an error message if it cannot locate the group in Active Directory. You must resolve this error before before proceeding.

j. Select **Save**.

5. Repeat Step 3 and Step 4 of this procedure for the other EPM System components in your deployment.
6. In the Change Center, click **Release Configuration**.
7. Restart WebLogic Server.

Enable Client Certificate-Based Authentication in Web Applications

Insert `login-config` definition in the configuration file of the following application archives located within *`EPM_ORACLE_HOME`*`/products/`.

- `Essbase/eas/server/AppServer/InstallableApps/Common/eas.ear`
- `FinancialDataQuality/AppServer/InstallableApps/aif.ear`
- `financialreporting/InstallableApps/HReports.ear`
- `Profitability/AppServer/InstallableApps/common/profitability.ear`

To enable client certificate-based authentication:

1. Stop EPM System components and Processes.
2. Using 7 Zip, expand a web archive contained within the enterprise archive; for example,*`EPM_ORACLE_HOME`*`/products/Essbase/eas/server/AppServer/InstallableApps/Common/eas.ear/eas.war`.
3. Navigate to `WEB-INF`.
4. Modify `web.xml` by adding the following `login_config` definition immediately before the `</webapp>` element:

```
<login-config>
     <auth-method>CLIENT-CERT</auth-method>
</login-config>
```

5. Save `web.xml`.
6. Click **Yes** when 7 Zip queries whether you want to update the archive.

Updating EPM System Security Configuration

Configure EPM System security to honor SSO. See Configuring EPM System for SSO.

Configuring EPM System for SSO

Oracle Enterprise Performance Management System products must be configured to support security agent for SSO. The configuration specified in Oracle Hyperion 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:

1. Launch the Oracle Hyperion Shared Services Console as a Shared Services Administrator. See Launching Shared Services Console.
2. Select **Administration**, and 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.

   For instructions, see Configuring User Directories.
4. Select **Security Options**.
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 the following table. Also, see Supported SSO Methods.

      > **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 3-6 Preferred SSO Mechanisms for Web Identity Management Solutions

Web Identity Management Solution	Recommended SSO Mechanism
Oracle Access Manager	`Custom HTTP Header`[1]
OSSO	`Custom HTTP Header`
SiteMinder	`Custom HTTP Header`
Kerberos	`Get Remote User from HTTP Request`

[1] The 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 Oracle Smart View for Office 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, some limitations exist:

- If Smart View is launched from an existing browser session that is connected to an Oracle Enterprise Performance Management System component, users must sign into Smart View again, because it does not share the cookie from the existing session.
- If you are using a custom HTML based login form instead of the default Oracle Access Manager login form, ensure that the source of the custom form includes the string `loginform`. This is required to allow Smart View integration with Oracle Access Manager to work.

4 Configuring User Directories

Related Topics

User Directories and EPM System Security

Oracle Enterprise Performance Management System products are supported on a number of user and identity management systems, which are collectively referred to as user directories. These include Lightweight Directory Access Protocol (LDAP) enabled user directories such as Sun Java System Directory Server (formerly SunONE Directory Server) and Active Directory. EPM System also supports relational databases as external user directories.

Generally, EPM System products use Native Directory and external user directories in provisioning. See Oracle Enterprise Performance Management System Certification Matrix for a list of supported user directories.

EPM System products require a user directory account for each user who accesses the products. These users may be assigned to groups to facilitate provisioning. Users and groups can be provisioned with EPM System roles and object ACLs. Because of the administrative overhead, Oracle does not recommend the provisioning of individual users. Users and groups from all configured user directories are visible from Oracle Hyperion Shared Services Console.

By default, EPM System Configurator configures the Shared Services repository as the Native Directory to support EPM System products. Directory Managers access and manage Native Directory using the Shared Services Console.

Operations Related to User Directory Configuration

To support SSO and authorization, System Administrators must configure external user directories. From Oracle Hyperion Shared Services Console, System Administrators can perform several tasks related to configuring and managing user directories. These topics provide instructions:

- Configuring user directories:
  - Configuring OID, Active Directory, and Other LDAP-based User Directories
  - Configuring Relational Databases as User Directories
- Testing User Directory Connections
- Editing User Directory Settings
- Deleting User Directory Configurations
- Managing the User Directory Search Order
- Setting Security Options

Oracle Identity Manager and EPM System

Oracle Identity Manager is a role and user administration solution that automates the process of adding, updating, and deleting both user accounts and attribute-level entitlements across enterprise resources. Oracle Identity Manager is available as a stand-alone product or as part of Oracle Identity and Access Management Suite Plus.

Oracle Enterprise Performance Management System integrates with Oracle Identity Manager by using enterprise roles which are LDAP groups. Roles of EPM System components can be assigned to enterprise roles. Users or groups added to Oracle Identity Manager enterprise roles automatically inherit assigned EPM System roles.

For example, assume that you have a Oracle Hyperion Planning application named *Budget Planning*. To support this application, you can create three enterprise roles—Budget Planning Interactive User, Budget Planning End User, and Budget Planning Admin—in Oracle Identity Manager. While provisioning EPM System roles, ensure that Provisioning Managers provision the enterprise roles from Oracle Identity Manager with the required roles from *Budget Planning* and other EPM System components including Shared Services. All users and groups assigned to the enterprise roles in Oracle Identity Manager inherits the EPM System roles. See Oracle Identity Manager documentation for information on deploying and managing Oracle Identity Manager.

To integrate Oracle Identity Manager with EPM System, Administrators must perform these steps:

- Ensure that members (users and groups) of Oracle Identity Manager enterprise roles that are to be used for EPM System provisioning are defined in an LDAP-enabled user directory; for example OID or Active Directory.
- Configure the LDAP-enabled user directory where members of the enterprise roles are defined as an external user directory in EPM System. See Configuring OID, Active Directory, and Other LDAP-based User Directories.

Active Directory Information

This section explains Microsoft Active Directory concepts used in this document.

DNS Lookup and Host Name Lookup

System Administrators can configure Active Directory so Oracle Hyperion Shared Services can perform a static host name lookup or a DNS lookup to identify Active Directory. Static host name lookup does not support Active Directory failover.

Using the DNS lookup ensures high availability of Active Directory in scenarios in which Active Directory is configured on multiple domain controllers to ensure high availability. When configured to perform a DNS lookup, Shared Services queries the DNS server to identify registered domain controllers and connects to the domain controller with the greatest weight. If the domain controller to which Shared Services is connected fails, Shared Services dynamically switches to the next available domain controller with the greatest weight.

> **Note:**
>
> DNS lookup can be configured only if a redundant Active Directory setup that supports failover is available. See Microsoft documentation for information.

Global Catalog

A global catalog is a domain controller that stores a copy of all Active Directory objects in a forest. It stores a complete copy of all objects in the directory for its host domain and a partial copy of all objects for all other domains in the forest, which are used in typical user search operations. See Microsoft documentation for information on setting up a global catalog.

If your organization is using a global catalog, use one of these methods to configure Active Directory:

- Configure the global catalog server as the external user directory (recommended).
- Configure each Active Directory domain as a separate external user directory.

Configuring the global catalog instead of individual Active Directory domains allows Oracle Enterprise Performance Management System products to access local and universal groups within the forest.

Configuring OID, Active Directory, and Other LDAP-based User Directories

System Administrators use the procedures in this section to configure LDAP-based corporate user directories, such as OID, Sun Java System Directory Server, Oracle Virtual Directory, Active Directory, IBM Tivoli Directory Server, or an LDAP-based user directory that is not listed on the configuration screen.

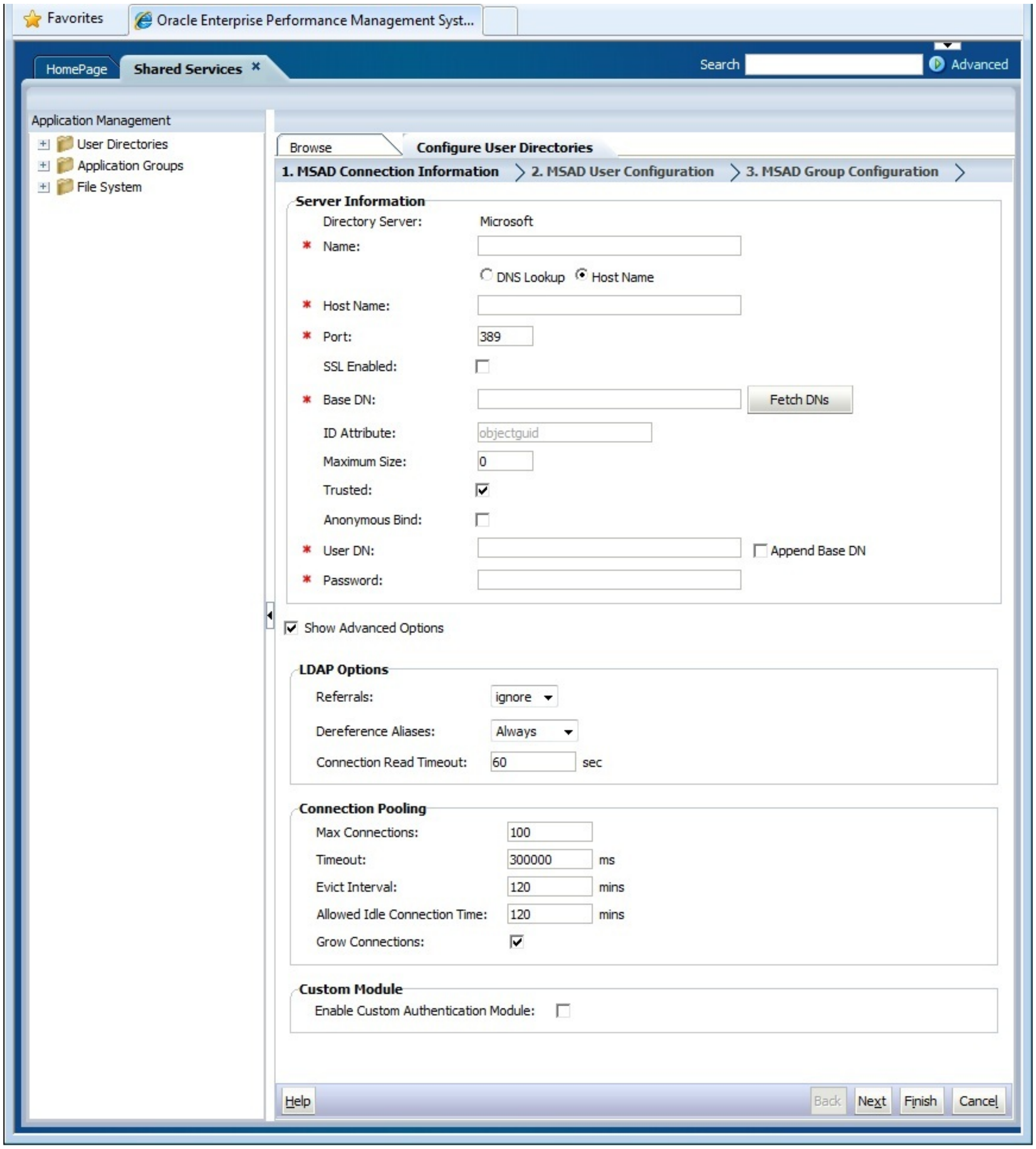
To configure OID, Active Directory, and other LDAP-based user directories:

1. Access Oracle Hyperion Shared Services Console as System Administrator. See Launching Shared Services Console.
2. Select **Administration**, and then **Configure User Directories**.

   The Provider Configuration tab opens. This screen lists all configured user directories, including Native Directory.
3. Click **New**.
4. Under **Directory Type**, select an option:
   - **Lightweight Directory Access Protocol (LDAP)** to configure an LDAP-based user directory other than Active Directory. Select this option to configure Oracle Virtual Directory.
   - **Microsoft Active Directory (MSAD)** to configure Active Directory.

Active Directory and Active Directory Application Mode (ADAM) only: If you want to use a custom ID attribute (an attribute other than `ObjectGUID`; for example `sAMAccountName` with Active Directory or ADAM, select **Lightweight Directory Access Protocol (LDAP)**, and configure it as Directory Type `Other`.

5. Click **Next**.


Favorites
Oracle Enterprise Performance Management Syst...
HomePage
Shared Services
Search
Advanced
Application Management
User Directories
Application Groups
File System
Browse
Configure User Directories
1. MSAD Connection Information
2. MSAD User Configuration
3. MSAD Group Configuration
Help
Back
Next
Finish
Cancel

Server Information

Directory Server: Microsoft

* Name:

○ DNS Lookup ● Host Name

* Host Name:

* Port: 389

SSL Enabled: □

* Base DN: Fetch DNs

ID Attribute: objectguid

Maximum Size: 0

Trusted: ☑

Anonymous Bind: □

* User DN: □ Append Base DN

* Password:

☑ Show Advanced Options

LDAP Options

Referrals: ignore

Dereference Aliases: Always

Connection Read Timeout: 60 sec

Connection Pooling

Max Connections: 100

Timeout: 300000 ms

Evict Interval: 120 mins

Allowed Idle Connection Time: 120 mins

Grow Connections: ☑

Custom Module

Enable Custom Authentication Module: □

6. Enter the required parameters.

Table 4-1 Connection Information Screen

Label	Description
Directory Server	Select a user directory. The **ID Attribute** value changes to the recommended constant unique identity attribute for the selected product. This property is automatically selected if you chose Active Directory in step 4. Select `Other` in the following scenarios: • You are configuring an unlisted user directory type; for example, Oracle Virtual Directory • You are configuring a listed LDAP-enabled user directory (for example, OID), but want to use a custom ID Attribute. • You are configuring Active Directory or ADAM to use a custom ID Attribute. **Note:** Because Oracle Virtual Directory provides a virtualized abstraction of LDAP directories and RDMBS data repositories in one directory view, Oracle Enterprise Performance Management System considers it a single external user directory regardless of the number and type of user directories Oracle Virtual Directory supports. **Example:** `Oracle Internet Directory`
Name	A descriptive name for the user directory. Used to identify a specific user directory if multiple user directories are configured. Name should not contain special characters other than space and underscore. **Example:** `Corporate_OID`
DNS Lookup	**Active Directory only:** Select this option to enable DNS lookup. See DNS Lookup and Host Name Lookup. Oracle recommends that you configure DNS lookup as the method to connect to Active Directory in production environments to avoid connection failures. **Note:** Do not select this option if you are configuring a global catalog. When you select this option, the following fields are displayed: • **Domain**: The domain name of an Active Directory forest. **Examples:** `example.com` or `us.example.com` • **AD Site**: Active Directory site name, generally the relative distinguished name of the site object that is stored in Active Directory configuration container. Typically, AD Site identifies a geographic location such as a city, state, region, or country. **Examples:** `Santa Clara` or `US_West_region` • **DNS Server**: DNS name of the server that supports DNS server lookup for domain controllers.

Table 4-1 (Cont.) Connection Information Screen

Label	Description
Host Name	**Active Directory only:** Select this option to enable static host name lookup. See DNS Lookup and Host Name Lookup. **Note:** Select this option if you are configuring an Active Directory global catalog.
Host Name	DNS name of the user directory server. Use the fully qualified domain name if the user directory is to be used to support SSO from SiteMinder. Oracle recommends using the host name to establish an Active Directory connection for testing purposes only. **Note:** If you are configuring an Active Directory global catalog, specify the global catalog server host name. See Global Catalog. **Example:** `MyServer`
Port	The port number where the user directory is running. **Note:** If you are configuring an Active Directory global catalog, specify the port used by the global catalog server (default is 3268). See Global Catalog. **Example:** `389`
SSL Enabled	The check box that enables secure communication with this user directory. The user directory must be configured for secure communication.
Base DN	The distinguished name (DN) of the node where the search for users and groups should begin. You can also use the **Fetch DNs** button to list available base DNs and then select the appropriate base DN from the list. **Note:** If you are configuring a global catalog, specify the base DN of the forest. See Using Special Characters for restrictions on the use of special characters. Oracle recommends that you select the lowest DN that contains all EPM System product users and groups. **Example:** `dc=example,dc=com`

Table 4-1 (Cont.) Connection Information Screen

Label	Description
ID Attribute	This attribute value can be modified only if `Other` is selected in **Directory Type**. This attribute must be a common attribute that exists in user and group objects on the directory server. The recommended value of this attribute is automatically set for OID `orclguid`, SunONE (`nsuniqueid`), IBM Directory Server (`Ibm-entryUuid`), Novell eDirectory (`GUID`), and Active Directory (`ObjectGUID`). **Example:** `orclguid` The ID attribute value, if you set it manually after choosing `Other` in **Directory Server**; for example to configure an Oracle Virtual Directory, should: • Point to a unique attribute • Not be location specific • Not change over time
Maximum Size	The maximum number of results that a search can return. If this value is greater than that supported by the user directory settings, the user directory value overrides this value. For user directories other than Active Directory, leave this field blank to retrieve all users and groups that meet the search criteria. For Active Directory, set this value to `0` to retrieve all users and groups that meet the search criteria. If you are configuring Oracle Hyperion Shared Services in Delegated Administration mode, set this value to 0.
Trusted	The check box to indicate that this provider is a trusted SSO source. SSO tokens from trusted sources do not contain the user's password.
Anonymous Bind	The check box to indicate that Shared Services can bind anonymously to the user directory to search for users and groups. Can be used only if the user directory allows anonymous binds. If this option is not selected, you must specify in the User DN an account with sufficient access permissions to search the directory where user information is stored. Oracle recommends that you not use anonymous bind. **Note:** Anonymous bind is not supported for OID.
User DN	This option is disabled if **Anonymous Bind** is selected. The distinguished name of the user that Shared Services should use to bind with the user directory. This user must have search privileges on the RDN attribute within the DN. For example, in the dn: `cn=John Doe, ou=people, dc=myCompany, dc=com`, the bind user should have search access to the `cn` attribute. Special characters in User DN must be specified using escape characters. See Using Special Characters for restrictions. **Example:** `cn=admin,dc=myCompany,dc=com`
Append Base DN	The check box for appending the base DN to the User DN. If you are using Directory Manager account as the User DN, do not append Base DN. This check box is disabled if the Anonymous Bind option is selected.

Table 4-1 (Cont.) Connection Information Screen

Label	Description
Password	User DN password This box is disabled if the Anonymous Bind option is selected. **Example:** `UserDNpassword`
Show Advanced Options	The check box to display advanced options.
Referrals	**Active Directory only:** If Active Directory is configured to follow referrals, select `follow` to automatically follow LDAP referrals. Select `ignore` to not use referrals.
Dereference Aliases	Select the method that Shared Services searches should use to dereference aliases in the user directory so searches retrieve the object to which the DN of the alias points. Select: • **Always**: Always dereference aliases. • **Never**: Never dereference aliases. • **Finding**: Dereference aliases only during name resolution. • **Searching**: Dereference aliases only after name resolution.
Connection Read Timeout	Interval (seconds) after which the LDAP provider aborts the LDAP read attempt if it does not get a response. **Default:** 60 seconds
Max Connections	Maximum connections in the connection pool. Default is 100 for LDAP-based directories, including Active Directory. **Default:** 100
Timeout	Timeout to get a connection from the pool. An exception is thrown after this period. **Default:** 300000 milliseconds (5 minutes)
Evict Interval	**Optional:** The interval for running the eviction process to clean the pool. The eviction process removes idle connections that have exceeded the `Allowed Idle Connection Time`. **Default:** 120 minutes
Allowed Idle Connection Time	**Optional:** The time after which the eviction process removes the idle connections in the pool. **Default:** 120 minutes
Grow Connections	This option indicates whether the connection pool can grow beyond `Max Connections`. Selected by default. If you do not allow the connection pool to grow, the system returns an error if a connection is not available within the time set for `Time Out`.
Enable Custom Authentication Module	The check box to enable the use of a custom authentication module to authenticate users defined in this user directory. You must also enter the fully qualified Java class name of the authentication module in the Security Options screen. See Setting Security Options. The custom authentication module authentication is transparent to thin and thick clients and does not require client deployment changes. See "Using a Custom Authentication Module" in the *Oracle Enterprise Performance Management System Security Configuration Guide*.

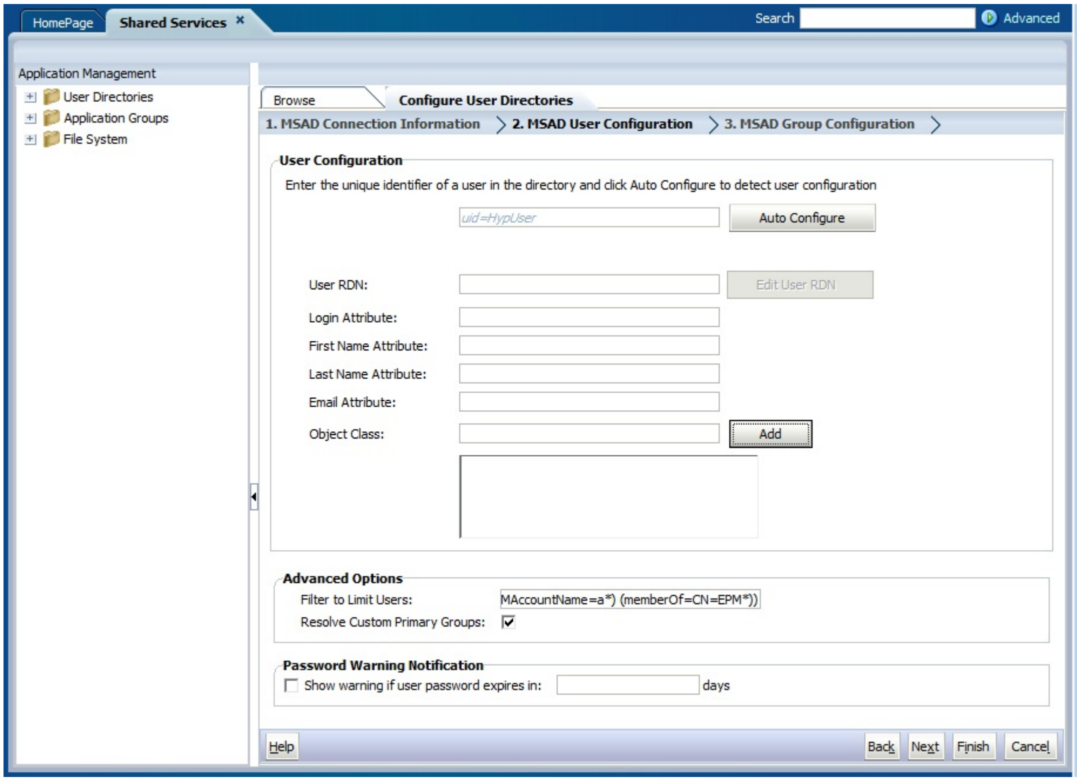
7. Click **Next**.

   Shared Services uses the properties set on the User Configuration screen to create a user URL that is used to determine the node where search for users begins. Using this URL speeds the search.

Caution:

The user URL should not point to an alias. EPM System security requires that the user URL point to an actual user.

Oracle recommends that you use the Auto Configure area of the screen to retrieve the required information.



User Configuration

Enter the unique identifier of a user in the directory and click Auto Configure to detect user configuration

uid=HypUser

User RDN:

Login Attribute:

First Name Attribute:

Last Name Attribute:

Email Attribute:

Object Class:

Advanced Options

Filter to Limit Users: MAccountName=a*) (memberOf=CN=EPM*))

Resolve Custom Primary Groups: ☑

Password Warning Notification

☐ Show warning if user password expires in: days

Note:

See Using Special Characters for a list of special characters that can be used in the user configuration.

8. In **Auto Configure**, enter a unique user identifier using the format `attribute=identifier`; for example, `uid=jdoe`.

   Attributes of the user are displayed in the User Configuration area.

   If you are configuring OID, you cannot automatically configure the user filter, because the root DSE of OID does not contain entries in the Naming Contexts attribute. See Managing Naming Contexts in the *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

> **Note:**
>
> You can manually enter required user attributes into text boxes in the User Configuration area.

Table 4-2 User Configuration Screen

Label	Description[1]
User RDN	The Relative DN of the user. Each component of a DN is called an RDN and represents a branch in the directory tree. The RDN of a user is generally the equivalent of the `uid` or `cn`. See Using Special Characters for restrictions. **Example:** `ou=People`
Login Attribute	A unique attribute (can be a custom attribute) that stores the login name of the user. Users use the value of this attribute as the user name while logging into EPM System products. User IDs (value of Login Attribute) must be unique across all user directories. For example, you may use `uid` and `sAMAccountName` respectively as the Login Attribute for your SunONE and Active Directory configurations. The values of these attributes must be unique across all user directories, including Native Directory. **Note:** User IDs are not case sensitive. **Note:** If you are configuring OID as an external user directory for EPM System products deployed on Oracle Application Server in a Kerberos environment, you must set this property to `userPrincipalName`. **Default** • **Active Directory:** `cn` • **LDAP directories other than Active Directory:** `uid`
First Name Attribute	The attribute that stores the user's first name **Default:** `givenName`
Last Name Attribute	The attribute that stores the user's last name **Default:** `sn`
Email Attribute	**Optional**: The attribute that stores the user's email address **Default:** `mail`

Table 4-2 (Cont.) User Configuration Screen

Label	Description[1]
Object Class	Object classes of the user (the mandatory and optional attributes that can be associated with the user). Shared Services uses the object classes listed in this screen in the search filter. Using these object classes, Shared Services should find all users who should be provisioned. **Note:** If you are configuring Active Directory or ADAM as user directory type `Other` to use a custom ID attribute, you must set this value to `user`. You can manually add object classes if needed. To add an object class, enter the object class name into the **Object Class** box, and then click **Add**. To delete object classes, select the object class and click **Remove**. **Default** - **Active Directory:** `user` - **LDAP directories other than Active Directory:** `person, organizationalPerson, inetorgperson`
Filter to Limit Users	An LDAP query that retrieves only the users that are to be provisioned with EPM System product roles. For example, the LDAP query `(uid=Hyp*)` retrieves only users whose names start with `Hyp`. The User Configuration screen validates the User RDN and recommends the use of a user filter, if required. The user filter limits the number of users returned during a query. It is especially important if the node identified by the user RDN contains many users that need not be provisioned. User filters can be designed to exclude the users that are not to be provisioned, thereby improving performance.

User Search Attribute for Multi-Attribute RDN	**LDAP-enabled user directories other than Active Directory only:** Set this value only if your directory server is configured to use a multi-attribute RDN. The value you set must be one of the RDN attributes. The value of the attribute you specify should be unique and the attribute should be searchable.
	For example, assume that a SunONE directory server is configured to combine the cn (`cn=John Doe`) and uid (`uid=jDoe12345`) attributes to create a multi-attribute RDN similar to the following:
	`cn=John Doe+uid=jDoe12345, ou=people,`
	` dc=myCompany, dc=com`
	In this case, you can use either `cn` or `uid` if these attributes meet the following conditions:
	- The attribute is searchable by the user identified in User DN filed on Connection Information tab
	- The attribute requires you to set a unique value across the user directory

Table 4-2 (Cont.) User Configuration Screen

Label	Description[1]
Resolve Custom Primary Groups	**Active Directory only:** The check box that indicates whether to identify primary groups of users to determine effective roles. This check box is selected by default. Oracle recommends that you not change this setting.
Show warning if user password expires in:	**Active Directory only:** The check box that indicates whether to display a warning message if the Active Directory user password expires within the specified number of days.

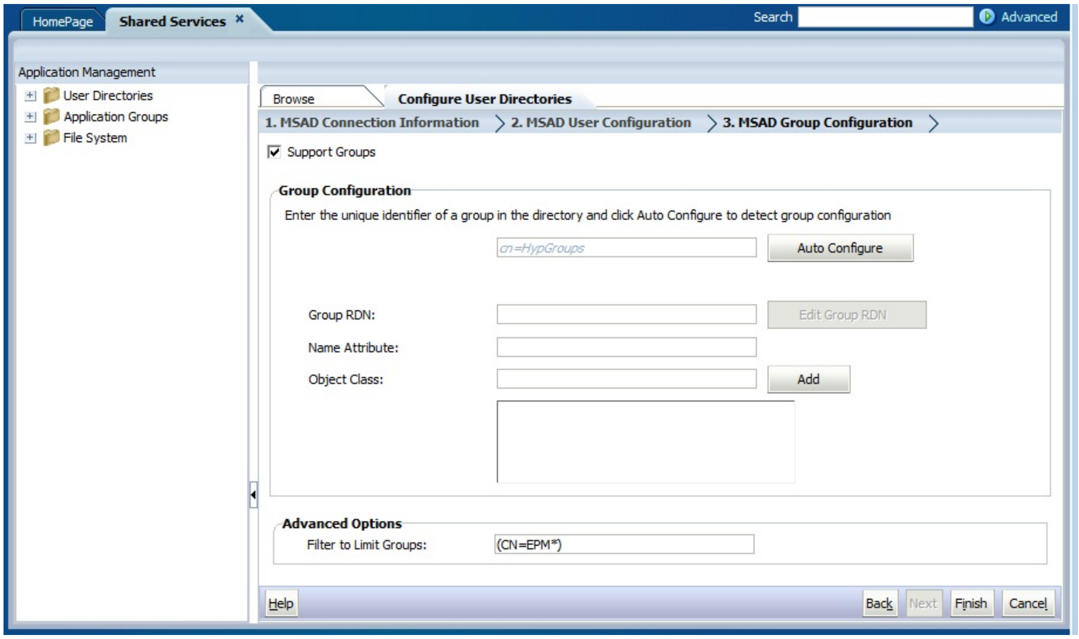
[1] EPM System security may use default values for some fields for which configuration value is optional. If you do not enter values in such fields, default values are used during runtime.

9. Click **Next**.

   The Group Configuration screen opens. Shared Services uses the properties set in this screen to create the group URL that determines the node where the search for groups starts. Using this URL speeds the search.

   > **Caution:**
   >
   > The Group URL should not point to an alias. EPM System security requires that the group URL point to an actual group. If you are configuring a Novell eDirectory that uses group aliases, the group aliases and group accounts must be available within the group URL.

   > **Note:**
   >
   > Data entry in the Group Configuration screen is optional. If you do not enter the group URL settings, Shared Services searches within the Base DN to locate groups, which can negatively affect performance, especially if the user directory contains many groups.



☑ Support Groups

Group Configuration

Enter the unique identifier of a group in the directory and click Auto Configure to detect group configuration

cn=HypGroups

Group RDN:

Name Attribute:

Object Class:

Advanced Options

Filter to Limit Groups: (CN=EPM*)

10. Clear **Support Groups** if your organization does not plan to provision groups, or if users are not categorized into groups on the user directory. Clearing this option disables the fields on this screen.

    If you are supporting groups, Oracle recommends that you use the autoconfigure feature to retrieve the required information.

    If you are configuring OID as a user directory, you cannot use the autoconfigure feature, because the root DSE of OID does not contain entries in the Naming Contexts attribute. See Managing Naming Contexts in the *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

11. In the **Auto Configure** text box, enter a unique group identifier, and then click **Go**.

    The group identifier must be expressed in `attribute=identifier` format; for example, `cn=western_region`.

    Attributes of the group are displayed in the Group Configuration area.

> **Note:**
>
> You can enter required group attributes in the Group Configuration text boxes.

> **Caution:**
>
> If the group URL is not set for user directories that contain / (slash) or \ (backslash) in its node names, the search for users and groups fails. For example, any operation to list the user or group fails if the group URL is not specified for a user directory in which users and groups exist in a node, such as `OU=child\ou,OU=parent/ou` or `OU=child/ou,OU=parent \ ou`.

Table 4-3   Group Configuration Screen

Label	Description[1]
Group RDN	The Relative DN of the group. This value, which is path relative to the Base DN, is used as the group URL. Specify a Group RDN that identifies the lowest user directory node in which all the groups that you plan to provision are available. If you use an Active Directory primary group for provisioning, ensure that the primary group falls under the Group RDN. Shared Services does not retrieve the primary group if it is outside the scope of the group URL. The Group RDN has a significant impact on login and search performance. Because it is the starting point for all group searches, you must identify the lowest possible node in which all groups for EPM System products are available. To ensure optimum performance, the number of groups present within the Group RDN should not exceed 10,000. If more groups are present, use a group filter to retrieve only the groups that you want to provision. **Note:** Shared Services displays a warning if the number of available groups within the Group URL exceeds 10,000. See Using Special Characters for restrictions. **Example:** `ou=Groups`
Name Attribute	The attribute that stores the name of the group **Default** • **LDAP directories including Active Directory:** `cn` • **Native Directory:** `cssDisplayNameDefault`

Table 4-3 (Cont.) Group Configuration Screen

Label	Description[1]
Object class	Object classes of the group. Shared Services uses the object classes listed in this screen in the search filter. Using these object classes, Shared Services should find all groups associated with the user. **Note:** If you are configuring Active Directory or ADAM as user directory type `Other` to use a custom ID attribute, you must set this value to `group?member`. You can manually add object classes if needed. To add an object class, enter the object class name into the Object class text box, and then click **Add**. To delete object classes, select the object class, and then click **Remove**. **Default** • **Active Directory:** `group?member` • **LDAP directories other than Active Directory:** `groupofuniquenames?uniquemember, groupOfNames?member` • **Native Directory:** `groupofuniquenames?uniquemember, cssGroupExtend?cssIsActive`
Filter to Limit Groups	An LDAP query that retrieves the groups that are to be provisioned with EPM System product roles only. For example, the LDAP query `(|(cn=Hyp*)(cn=Admin*))` retrieves only groups whose names start with `Hyp` or `Admin`. The group filter, used to limit the number of groups returned during a query, is especially important if the node identified by the Group RDN contains a large number of groups that need not be provisioned. Filters can be designed to exclude the groups that are not to be provisioned, improving performance. If you use Active Directory primary group for provisioning, ensure that any group filter that you set can retrieve the primary group contained within the scope of the group URL. For example, the filter `(|(cn=Hyp*)(cn=Domain Users))` retrieves groups that have names that start with `Hyp` and the primary group named `Domain Users`.

[1] EPM System security may use default values for some fields for which configuration value is optional. If you do not enter values in such fields, default values are used during runtime.

12. Click **Finish**.

    Shared Services saves the configuration and returns to the Defined User Directories screen, which now lists the user directory that you configured.

13. Test the configuration. See Testing User Directory Connections.
14. If needed, change the search order assignment. See Managing the User Directory Search Order for details.
15. If needed, specify security options. See Setting Security Options for details.
16. Restart Oracle Hyperion Foundation Services and other EPM System components.

Configuring Relational Databases as User Directories

User and group information from the system tables of Oracle, SQL Server, and IBM DB2 relational databases can be used to support provisioning. If group information cannot be derived from the database's system schema, Oracle Hyperion Shared Services does not support the provisioning of groups from that database provider. For example, Shared Services cannot extract group information from older versions of IBM DB2, because the database uses groups defined on the operating system. Provisioning Managers can, however, add these users to groups in Native Directory and provision those groups. For supported platform information, see the *Oracle Enterprise Performance Management System Certification Matrix* posted on the Oracle Fusion Middleware Supported System Configurations page on Oracle Technology Network (OTN).

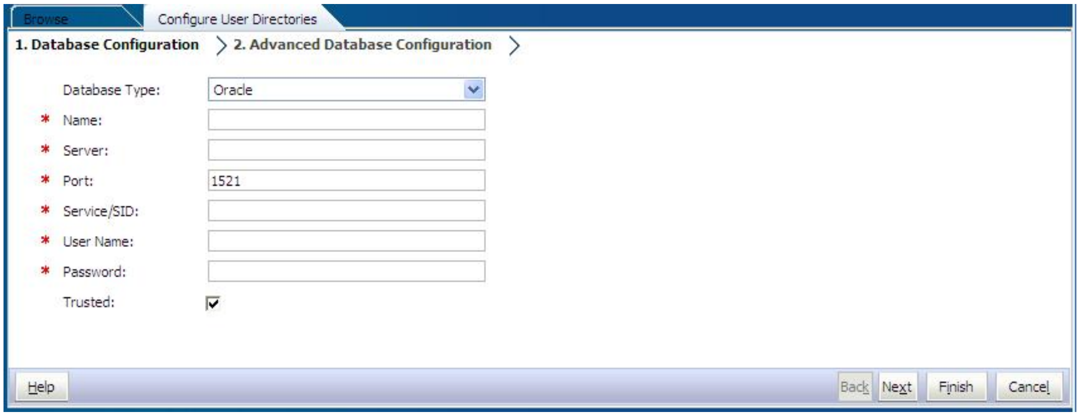
> **Note:**
>
> If you are using a DB2 database, the user name must contain at least eight characters. User names should not exceed 256 characters (Oracle and SQL Server databases), and 1000 characters (DB2).

Configure Shared Services to connect to the database as the database administrator; for example, Oracle `SYSTEM` user, to retrieve the list of users and groups.

> **Note:**
>
> Shared Services retrieves only active database users for provisioning. Inactive and locked database user accounts are ignored.

To configure database providers:

1. Access Oracle Hyperion Shared Services Console as System Administrator. See Launching Shared Services Console.
2. Select **Administration**, and then **Configure User Directories**.
3. Click **New**.
4. In the **Directory Type** screen, select **Relational Database (Oracle, DB2, SQL Server)**.
5. Click **Next**.


Browse
Configure User Directories
1. Database Configuration
2. Advanced Database Configuration
Help
Back
Next
Finish
Cancel

Database Type: Oracle

* Name:

* Server:

* Port: 1521

* Service/SID:

* User Name:

* Password:

Trusted: ☑

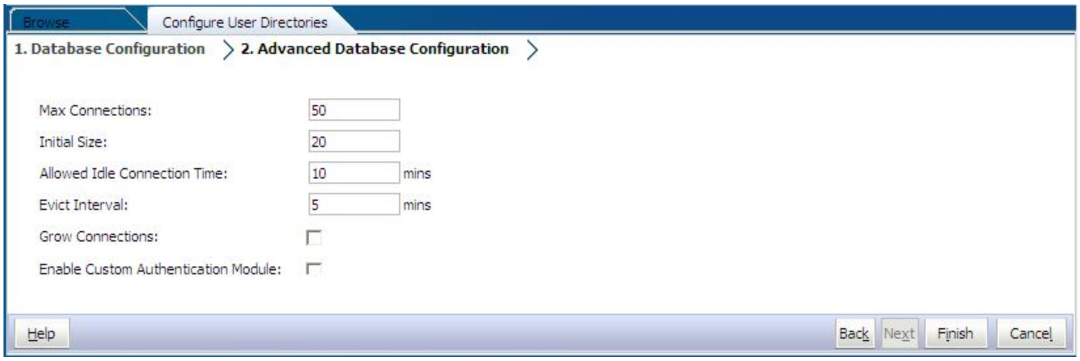
6. On the Database Configuration tab, enter configuration parameters.

Table 4-4 Database Configuration Tab

Label	Description
Database Type	The relational database provider. Shared Services supports only Oracle and SQL Server databases as database providers. **Example:** `Oracle`
Name	A unique configuration name for the database provider. **Example:** `Oracle_DB_FINANCE`
Server	The DNS name of the computer on which the database server is running. **Example:** `myserver`
Port	The database server port number **Example:** `1521`
Service/SID (Oracle only)	The system identifier (default is `orcl`) **Example:** `orcl`
Database (SQL Server and DB2 only)	The database to which Shared Services should connect **Example:** `master`
User Name	The user name that Shared Services should use to access the database. This database user must have access privileges to database system tables. Oracle recommends that you use the `system` account for Oracle databases and the database administrator's user name for SQL Server databases. **Example:** `SYSTEM`
Password	The password of the user identified in the **User Name**. **Example:** `system_password`
Trusted	The check box that specifies that this provider is a trusted SSO source. SSO tokens from trusted sources do not contain the user's password.

7. **Optional:** Click **Next** to configure the connection pool.

   The Advanced Database Configuration tab opens.


Browse
Configure User Directories
1. Database Configuration
2. Advanced Database Configuration
Help
Back
Next
Finish
Cancel

Max Connections: 50

Initial Size: 20

Allowed Idle Connection Time: 10 mins

Evict Interval: 5 mins

Grow Connections: □

Enable Custom Authentication Module: □

8. On Advanced Database Configuration, enter connection pool parameters.

Table 4-5   Advanced Database Configuration Tab

Label	Description
Max Connections	Maximum connections in the pool. Default is 50.
Initial Size	Available connections when the pool is initialized. Default is 20.
Allowed Idle Connection Time	**Optional:** The time after which the eviction process removes the idle connections in the pool. Default is 10 minutes.
Evict Interval	**Optional:** The interval for running the eviction process to clean up the pool. Eviction removes idle connections that have exceeded the `Allowed Idle Connection Time`. Default is five minutes.
Grow Connections	Indicates whether the connection pool can grow beyond `Max Connections`. By default, this option is cleared, indicating that the pool cannot grow. If you do not allow the connection pool to grow, the system returns an error if a connection is not available within the time set for `Time Out`.
Enable Custom Authentication Module	The check box to enable the use of a custom authentication module to authenticate users defined in this user directory. You must also enter the fully qualified Java class name of the authentication module in the Security Options screen. See Setting Security Options. The custom authentication module authentication is transparent to thin and thick clients. See "Using a Custom Authentication Module" in the *Oracle Enterprise Performance Management System Security Configuration Guide*.

9. Click **Finish**.
10. Click **OK** to return to the Defined User Directories screen.
11. Test the database provider configuration. See Testing User Directory Connections.
12. Change the search order assignment, if needed. See Managing the User Directory Search Order for details.
13. Specify security settings, if needed. See Setting Security Options.
14. Restart Oracle Hyperion Foundation Services and other Oracle Enterprise Performance Management System components.

Testing User Directory Connections

After configuring a user directory, test the connection to ensure that Oracle Hyperion Shared Services can connect to the user directory using the current settings.

To test a user directory connection:

1. Access Oracle Hyperion Shared Services Console as System Administrator. See Launching Shared Services Console.
2. Select **Administration**, and then **Configure User Directories**.
3. From the list of user directories, select an external user directory configuration to test.
4. Click **Test**, and then **OK**.

Editing User Directory Settings

Administrators can modify any parameter, other than the name, of a user directory configuration. Oracle recommends that you not edit the configuration data of user directories that were used for provisioning.

> **Caution:**
>
> Editing some settings, for example, the `ID Attribute`, in the user directory configuration invalidates provisioning data. Exercise extreme care when modifying the settings of a user directory that has been provisioned.

To edit a user directory configuration:

1. Access Oracle Hyperion Shared Services Console as System Administrator. See Launching Shared Services Console.
2. Select **Administration**, and then **Configure User Directories**.
3. Select a user directory to edit.
4. Click **Edit**.
5. Modify the configuration settings.

   > **Note:**
   >
   > You cannot modify the configuration name. If you are modifying an LDAP user directory configuration, you can choose a different directory server or `Other` (for custom LDAP directories) from the Directory Server list. You cannot edit Native Directory parameters.

   For an explanation of the parameters that you can edit, see the following tables:

   - Active Directory and other LDAP-based user directories, see the tables in Configuring OID, Active Directory, and Other LDAP-based User Directories.
   - Databases: See the table in Configuring Relational Databases as User Directories
6. Click **OK** to save the changes.

Deleting User Directory Configurations

System Administrators can delete an external user directory configuration anytime. Deleting a configuration invalidates all the provisioning information for the users and groups derived from the user directory and removes the directory from the search order.

> **Tip:**
>
> If you do not want to use a configured user directory that was used for provisioning, remove it from the search order so that it is not searched for users and groups. This action maintains the integrity of provisioning information and enables you to use the user directory later.

To delete a user directory configuration:

1. Access Oracle Hyperion Shared Services Console as System Administrator. See Launching Shared Services Console.
2. Select **Administration**, and then **Configure User Directories**.
3. Select a directory.
4. Click **Delete**.
5. Click **OK**.
6. Click **OK** again.
7. Restart Oracle Hyperion Foundation Services and other Oracle Enterprise Performance Management System components.

Managing the User Directory Search Order

When a System Administrator configures an external user directory, Oracle Hyperion Shared Services automatically adds the user directory to the search order and assigns it the next available search sequence preceding that of Native Directory. The search order is used to cycle through configured user directories when Oracle Enterprise Performance Management System searches for users and groups.

System Administrators can remove a user directory from the search order, in which case Shared Services automatically reassigns the search order of the remaining directories. User directories not included in the search order are not used to support authentication and provisioning.

> **Note:**
>
> Shared Services terminates the search for the user or group when it encounters the specified account. Oracle recommends that the corporate directory that contains most of the EPM System users be placed at the top of the search order.

By default, Native Directory is set as the last directory in the search order. Administrators can perform these tasks to manage the search order:

- Adding a User Directory to the Search Order
- Changing the Search Order
- Removing a Search Order Assignment

Adding a User Directory to the Search Order

A newly configured user directory is automatically added to the search order. If you removed a directory from the search order, you can add it to the end of the search order.

To add a user directory to the search order:

1. Access Oracle Hyperion Shared Services Console as System Administrator. See Launching Shared Services Console.
2. Select **Administration**, and then **Configure User Directories**.
3. Select a deactivated user directory to add to the search order.
4. Click **Include**.

   This button is available only if you have selected a user directory that is not in the search order.
5. Click **OK** to return to the Defined User Directories screen.
6. Restart Oracle Hyperion Foundation Services and other EPM System components.

Removing a Search Order Assignment

Removing a user directory from the search order does not invalidate the directory configuration; it removes the user directory from the list of directories that are searched for authenticating users. A directory that is not included in the search order is set to `Deactivated` status. When an Administrator removes a user directory from the search order, the search sequence assigned to the other user directories is automatically updated.

> **Note:**
>
> Native Directory cannot be removed from the search order.

To remove a user directory from the search order:

1. Access Shared Services Console as System Administrator. See Launching Shared Services Console.
2. Select **Administration**, and then **Configure User Directories**.
3. Select a directory to remove from the search order.
4. Click **Exclude**.
5. Click **OK**.
6. Click **OK** on the Directory Configuration Result screen.
7. Restart Foundation Services and other EPM System components.

Changing the Search Order

The default search order assigned to each user directory is based on the sequence in which the directory was configured. By default, Native Directory is set as the last directory in the search order.

To change the search order:

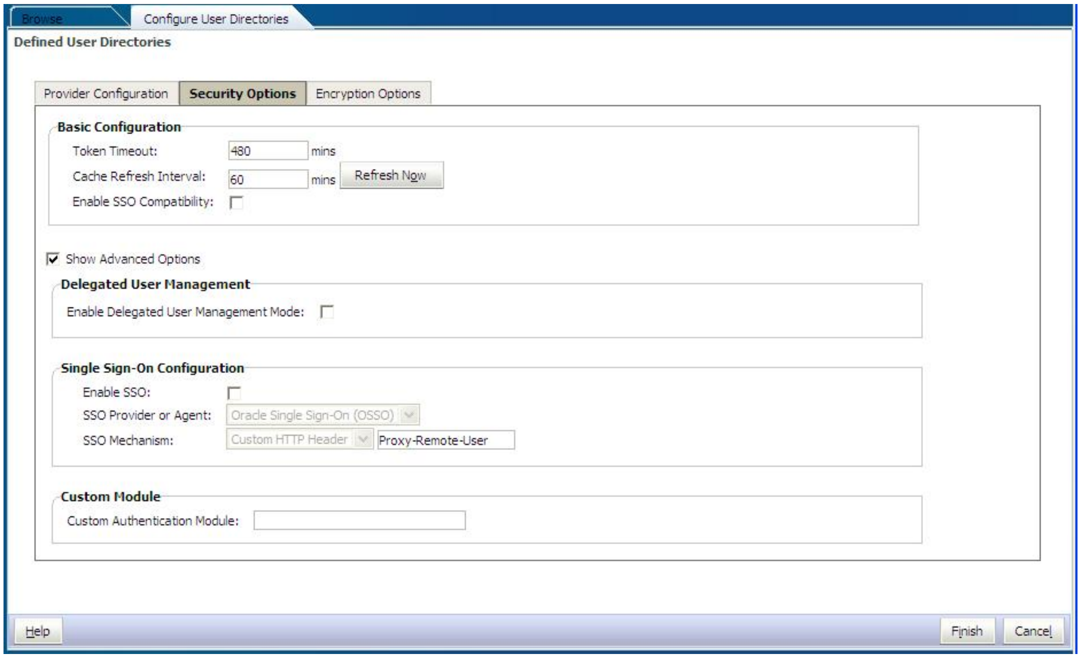
1. Access Shared Services Console as System Administrator. See Launching Shared Services Console.
2. Select **Administration**, and then **Configure User Directories**.
3. Select a directory whose search order you want to change.
4. Click **Move Up** or **Move Down**.
5. Click **OK**.
6. Restart Foundation Services, other EPM System components, and custom applications that use the Shared Services security APIs.

Setting Security Options

Security options comprise the global parameters applicable to all user directories included in the search order.

To set security options:

1. Access Oracle Hyperion Shared Services Console as System Administrator. See Launching Shared Services Console.
2. Select **Administration**, and then **Configure User Directories**.
3. Select **Security Options**.
4. In **Security Options**, set global parameters.



Basic Configuration

Token Timeout: 480 mins

Cache Refresh Interval: 60 mins

Enable SSO Compatibility: □

☑ Show Advanced Options

Delegated User Management

Enable Delegated User Management Mode: □

Single Sign-On Configuration

Enable SSO: □

SSO Provider or Agent: Oracle Single Sign-On (OSSO)

SSO Mechanism: Custom HTTP Header Proxy-Remote-User

Custom Module

Custom Authentication Module:

Table 4-6 Security Options for User Directories

Parameter	Description
Token Timeout	Time (in minutes) after which the SSO token issued by Oracle Enterprise Performance Management System products or the web identity management solution expires. Users must log in again after this period. Token timeout is set based on the server's system clock. Default is 480 minutes. **Note:** Token timeout is not the same as session timeout.
Cache Refresh Interval	Interval (in minutes) for refreshing the Oracle Hyperion Shared Services cache of groups to users relationship data. Default is `60` minutes. Shared Services caches information about new external user directory groups and new users added to existing groups only after the next cache refresh. Users provisioned through a newly created external user directory group do not get their provisioned roles until the cache is refreshed.
Refresh Now	Click this button to manually initiate the refreshing of Shared Services cache that contains groups to users relationship data. You may want to initiate a cache refresh after creating new groups in external user directories and provisioning them or after adding new users to existing groups. The cache is refreshed only after Shared Services makes a call that uses the data in the cache.
Enable SSO Compatibility	Select this option if your deployment is integrated with Oracle Business Intelligence Enterprise Edition Release 11.1.1.5 or earlier.
Enable Delegated User Management Mode	Option enabling delegated user management of EPM System products to support the distributed management of provisioning activities. See "Delegated User Management" in the *Oracle Enterprise Performance Management System User Security Administration Guide*.
Enable SSO	Option enabling support for SSO from security agents such as Oracle Access Manager
SSO Provider or Agent	Select the web identity management solution from which EPM System products should accept SSO. Select **Other** if your web identity management solution; for example, Kerberos, is not listed. The preferred SSO mechanism and name are automatically selected when you select the SSO provider. You can change the name of the SSO mechanism (HTTP header or custom login class), if required. If you select `Other` as the SSO provider or agent, you must censure that it supports an EPM System supported SSO mechanism. See "Supported SSO Methods" in the *Oracle Enterprise Performance Management System Security Configuration Guide*.

Table 4-6 (Cont.) Security Options for User Directories

Parameter	Description
SSO Mechanism	The method that the selected web identity management solution uses to provide user's login name to EPM System products. For a description of acceptable SSO methods, see "Supported SSO Methods" in the *Oracle Enterprise Performance Management System Security Configuration Guide*. • `Custom HTTP Header`: Set the name of the header that the security agent passes to EPM System. • `Custom Login Class`: Specify the custom Java class that handles HTTP requests for authentication. See "Custom Login Class" in the *Oracle Enterprise Performance Management System Security Configuration Guide*. **Note:** Custom Login Class is not the same as custom authentication. • `HTTP Authorization Header`: The standard HTTP mechanism. • `Get Remote User from HTTP Request`: Select this option if the security agent populates the remote user in the HTTP request.
Custom Authentication Module	The fully qualified Java class name of the custom authentication module (for example, `com.mycompany.epm.CustomAuthenticationImpl`) that should be used to authenticate users on all user directories for which the custom authentication module is selected. The authentication module is used for a user directory only if the directory configuration has enabled (default) its use. Oracle Hyperion Foundation Services requires that the custom authentication JAR file be named `CustomAuth.jar`. `CustomAuth.jar` must be available in `MIDDLEWARE_HOME\user_projects\domains\WEBLOGIC_DOMAIN\lib`, typically, `C:\Oracle/Middleware/user_projects/domains/EPMSystem/lib`. On all client installations, `CustomAuth.jar` must be present in `EPM_ORACLE_HOME/common/jlib/11.1.2.0`, typically, `C:\Oracle\Middleware\EPMSystem11R1\common\jlib\11.1.2.0`. You can use any package structure and class name within the JAR file. For more information, see "Using a Custom Authentication Module" in the *Oracle Enterprise Performance Management System Security Configuration Guide*.

5. Click **OK**.
6. Restart Foundation Services and other EPM System components.

Regenerating Encryption Keys

Oracle Enterprise Performance Management System uses the following keys to ensure security:

- Single Sign On Token encryption key, used to encrypt and decrypt EPM System SSO tokens. This key is stored in Oracle Hyperion Shared Services Registry
- Trusted Services key, used by EPM System components to verify the authenticity of the service that is requesting an SSO token
- Provider Configuration encryption key, used to encrypt the password (user DN password for LDAP-enabled user directories) that EPM System security uses to bind with a configured external user directory. This password is set while configuring an external user directory.

Change these keys periodically to strengthen EPM System security. Oracle Hyperion Shared Services and the security subsystem of EPM System use AES encryption with 128-bit key strength.

> **Caution:**
>
> Taskflows used by Oracle Hyperion Financial Management and Oracle Hyperion Profitability and Cost Management are invalidated when you regenerate the Single Sign On Encryption key. After regenerating the key, open and save the taskflows to revalidate them.

To regenerate the Single Sign On Encryption key, Provider Configuration key, or Trusted Services key:

1. Access Oracle Hyperion Shared Services Console as System Administrator. See Launching Shared Services Console.
2. Select **Administration**, and then **Configure User Directories**.
3. Select **Encryption Options**.
4. In **Encryption Options**, select the key that you want to regenerate.

Table 4-7   EPM System Encryption Options

Option	Description
Single Sign On Token	Select to regenerate the encryption key that is used to encrypt and decrypt EPM System SSO tokens. Select one of the following buttons if **Enable SSO Compatibility** is selected on **Security Options**: • **Generate new key** to create a new SSO token encryption key • **Reset to default** to restore the default SSO token encryption key **Note:** If you revert to the default encryption key, you must delete the existing keystore file (*EPM_ORACLE_HOME*`/common/CSS/ssHandlerTK`) from all EPM System host machines.
Trusted Services Key	Select this option to regenerate the trusted authentication key, used by EPM System components to verify the authenticity of the service that is requesting an SSO token.

Table 4-7 (Cont.) EPM System Encryption Options

Option	Description
Provider Configuration Key	Select this option to regenerate the key that is used to encrypt the password (user DN password for LDAP-enabled user directories) that EPM System security uses to bind with a configured external user directory. This password is set while configuring an external user directory.

5. Click **OK**.
6. If you chose to generate a new SSO encryption key, complete this step.
   a. Click **Download**.
   b. Click **OK** to save `ssHandlerTK`, the keystore file that supports the new SSO encryption key, into a folder on the server that hosts Oracle Hyperion Foundation Services.
   c. Copy `ssHandlerTK` into *EPM_ORACLE_HOME*`/common/CSS` on all EPM System host machines.
7. Restart Foundation Services and other EPM System components.

Using Special Characters

Active Directory and other LDAP-based user directories allow special characters in entities such as DNs, user names, roles, and group names. Special handling may be required for Oracle Hyperion Shared Services to understand such characters.

Generally, you must use escape characters while specifying special characters in user directory settings; for example, Base DN and user and group URLs. The following table lists the special characters that can be used in user names, group names, user URLs, group URLs, and in the value of OU in user DN.

Table 4-8 Supported Special Characters

Character	Name or Meaning	Character	Name or Meaning
(	open parenthesis	$	dollar
)	close parenthesis	+	plus
"	quotation mark	&	ampersand
'	single quotation mark	\	backslash
,	comma	^	caret
=	equal to	;	semicolon
<	less than	#	pound
>	greater than	@	at

Note:

Do not use / (slash) in organization unit names that come within the Base DN

- Special characters are not permitted in the value of the Login User attribute.
- The asterisk (*) is not supported in user names, group names, user and group URLs, and in the name of the OU in User DN.

- Attribute values containing a combination of special characters are not supported.
- The ampersand (&) can be used without an escape character. For Active Directory settings, & must be specified as `&amp;`.
- User and group names cannot contain both a backslash (\) and slash (/). For example, names such as `test/\user` and `new\test/user` are not supported.

Table 4-9 Characters that Need Not be Escaped

Character	Name or Meaning	Character	Name or Meaning
`(`	open parenthesis	'	single quote
`)`	close parenthesis	^	caret
`$`	dollar	@	at
`&`	Ampersand		

> **Note:**
>
> `&` must be stated as `&amp;`.

These characters must be escaped if you use them in user directory settings (user names, group names, user URLs, group URLs and User DN).

Table 4-10 Escape for Special Characters in User Directory Configuration Settings

Special Character	Escape	Example Setting	Escaped Example
comma (`,`)	backslash (`\`)	`ou=test,ou`	`ou=test\,ou`
plus sign (`+`)	backslash (`\`)	`ou=test+ou`	`ou=test\+ou`
equal to (`=`)	backslash (`\`)	`ou=test=ou`	`ou=test\=ou`
pound (`#`)	backslash (`\`)	`ou=test#ou`	`ou=test\#ou`
semicolon (`;`)	backslash (`\`)	`ou=test;ou`	`ou=test\;ou`
less than (`<`)	backslash (`\`)	`ou=test<ou`	`ou=test\<ou`
greater than (`>`)	backslash (`\`)	`ou=test>ou`	`ou=test\>ou`
quotation mark (`"`)	two backslashes (`\\`)	`ou=test"ou`	`ou=test\\"ou`
backslash (`\`)	three backslashes (`\\\`)	`ou=test\ou`	`ou=test\\\ou`

> **Note:**
>
> - In User DNs, quotation mark (`"`) must be escaped with one backslash. For example, `ou=test"ou` must be specified as `ou=test\"ou`.
> - In User DNs, a backslash (\) must be escaped with one backslash. For example, `ou=test\ou` must be specified as `ou=test\\ou`.

> **Caution:**
>
> If the user URL is unspecified, users created within the RDN root must not contain / (slash) or \ (backslash). Similarly, these characters should not be used in the names of groups created within the RDN root if a group URL is not specified. For example, group names such as `OU=child\ou,OU=parent/ou` or `OU=child/ou,OU=parent\ou` are not supported. This issue does not apply if you are using a unique attribute as the `ID Attribute` in the user directory configuration.

Special Characters in Native Directory

special characters are supported in user and group names in Native Directory.

Table 4-11   Supported Special Characters: Native Directory

Character	Name or Meaning	Character	Name or Meaning
@	at	,	comma
#	pound	=	equal to
$	dollar	+	plus
^	caret	;	semicolon
(	open parenthesis	!	exclamation
)	close parenthesis	%	percent
'	single quotation mark		

5 Using a Custom Authentication Module

Related Topics

- Overview
- Use-Case Examples and Limitations
- Prerequisites
- Design and Coding Considerations
- Deploying the Custom Authentication Module

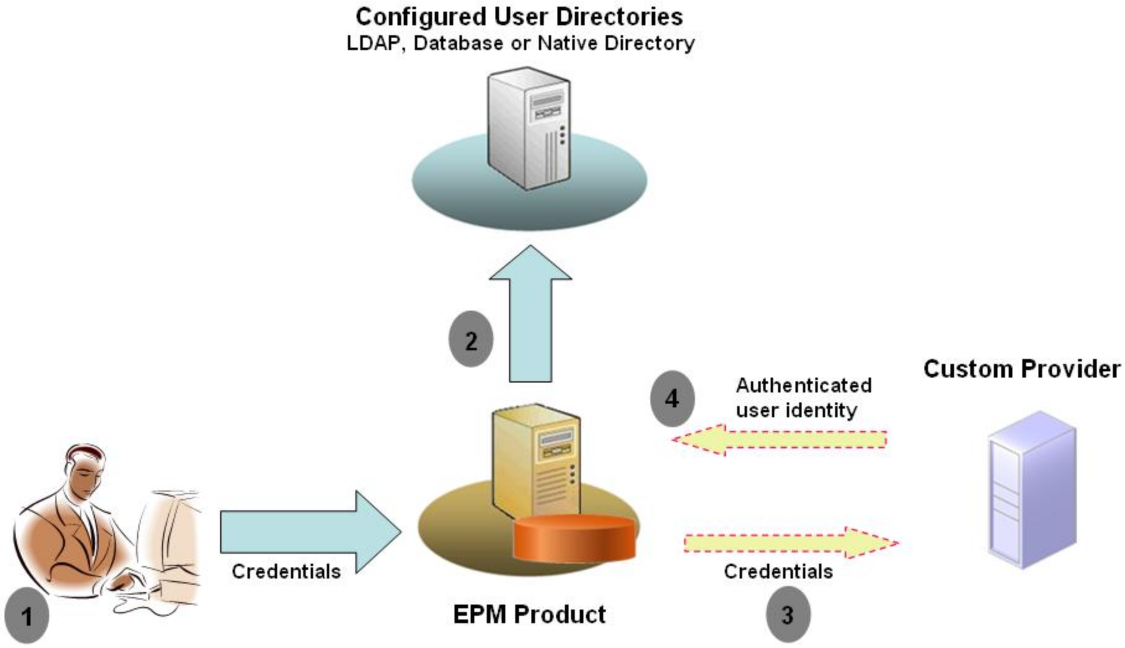
Overview

A custom authentication module is a Java module that customers develop and implement to authenticate Oracle Enterprise Performance Management 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 Smart View for Office, and Oracle Essbase Studio) and thin clients (for example, Oracle Hyperion Enterprise Performance Management 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:


Configured User Directories
LDAP, Database or Native Directory
2
4
Authenticated user identity
Custom Provider
1
Credentials
EPM Product
Credentials
3

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), 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 a user directory 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 onetime password Support
- Performing authentication against a Resource Access Control Facility (RACF)
- Adding a 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, Oracle Hyperion Enterprise Performance Management 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

- A fully tested Java archive named `CustomAuth.jar` that contains custom authentication module libraries. `CustomAuth.jar` must implement the public interface `CSSCustomAuthenticationIF`, defined in `com.hyperion.css` package as a part of the standard Oracle Hyperion Shared Services APIs. See http://download.oracle.com/docs/cd/E12825_01/epm.111/epm_security_api_11111/client/com/hyperion/css/CSSCustomAuthenticationIF.html.
- Access to Shared Services as Shared Services administrator

Design and Coding Considerations

Search Order

In addition to Native Directory, multiple user directories can be configured in Oracle Hyperion Shared Services. A default search order position is assigned to all configured user directories. You can modify the search order from Oracle Hyperion Shared Services Console. Excepting Native Directory, you can remove configured user directories from the search order. Oracle Enterprise Performance Management System does not use the user directories that are not included in the search order. See the *Oracle Enterprise Performance Management System User Security Administration 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
- Use-Case Scenario 2
- Use-Case Scenario 3

Use-Case Scenario 1

The following table 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 5-1 Setup for Scenario 1

User Directory Type and Name	Search Order	Custom Authentication	Sample User Names	Password[1]

Native Directory	1	Disabled	`test_user_1`
			`test_user_2`
			`test_user_3`
				`password`

LDAP-Enabled
SunONE_West
	2	Disabled	`test_ldap1`
			`test_ldap_2`
			`test_user_3`
			`test_ldap_4`
				`ldappassword`

LDAP-Enabled
SunONE_East
	3	Enabled	`test_ldap1`
			`test_ldap_2`
			`test_user_3`
				`ldappassword` on SunONE and `RSA PIN` in custom module

[1] For 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 5-2 User interaction and results

User Name and Password	Authentication Result	Login User Directory
`test_user_1/password`	Success	Native Directory
`test_user_3/password`	Success	Native Directory
`test_user_3/ldappassword`	Success	SunONE_West (search order 2)[1]
`test_user_3/RSA PIN`	Success	SunONE_East (search order 3)[2]
`test_ldap_2/ldappassword`	Success	SunONE_West (search order 2)

Table 5-2 (Cont.) User interaction and results

User Name and Password	Authentication Result	Login User Directory
`test_ldap_4/RSA PIN`	Failure EPM System displays an authentication error.[3]	

[1] The 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).

[2] EPM 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.

[3] Oracle 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

The following table 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_ldap_2`, to EPM System security

Table 5-3 A sample search order

User Directory	Search Order	Custom Authentication	Sample User Names	Password[1]

Native Directory	1	Disabled	`test_user_1`
			`test_user_2`
			`test_user_3`
				`password`

LDAP-Enabled, for example, SunONE	2	Enabled	`test_ldap1`
			`test_ldap2`
			`test_user_3`
				`ldappassword` on SunONE and `RSA PIN` in custom module

[1] For 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 5-4 User interaction and results

User Name and Password	Login Result	Login User Directory
`test_user_1/password`	Success	Native Directory
`test_user_3/password`	Success	Native Directory
`test_user_3/ldappassword`	Failure	SunONE[1]
`test_user_3/RSA PIN`	Success	SunONE[2]

[1] Authentication 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.

[2] Authentication 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

The following table 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 5-5 A sample search order

User Directory	Search Order	Custom Authentication	Sample User Names	Password[1]

Native Directory	1	Enabled	`test_user_1`
			`test_user_2`
			`test_user_3`
				`RSA_PIN`

LDAP-Enabled, for example, MSAD	2	Disabled	`test_ldap1`
			`test_ldap4`
			`test_user_3`
				`ldappassword`

LDAP-Enabled, for example, SunONE	3	Enabled	`test_ldap1`
			`test_ldap4`
			`test_user_3`
				`ldappassword` on SunONE and `RSA PIN` in custom module

[1] For 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 5-6 User interaction and results

User Name and Password	Authentication Result	Login User Directory
`test_user_1/password`	Success	Native Directory
`test_user_3/RSA_PIN`	Success	Native Directory
`test_user_3/ldappassword`	Success	MSAD (search order 2)
`test_ldap_4/ldappassword`	Success	MSAD (search order 2)
`test_ldap_4/RSA PIN`	Success	SunONE (search order 3)

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). 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 unsuccessful. 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 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

Only one custom module is supported for an Oracle Enterprise Performance Management 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

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 Oracle Hyperion Shared Services and any systems that use Shared Services APIs.
- Copy the custom authentication module Java archive `CustomAuth.jar` into the deployment:
  - **WebLogic:** Copy `CustomAuth.jar` into *`MIDDLEWARE_HOME`*`/user_projects/domains/`*`WEBLOGIC_DOMAIN`*`/lib`, typically, `C:/Oracle/Middleware/user_projects/domains/EPMSystem/lib`.

    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 *`MIDDLEWARE_HOME`*`/user_projects/domains/`*`WEBLOGIC_DOMAIN`*`/lib`.
  - **All Client Deployments**: Copy `CustomAuth.jar` into all EPM System client deployments, into the following location:

    EPM_ORACLE_HOME`/common/jlib/11.1.2.0`, typically, `Oracle/Middleware/common/jlib/11.1.2.0`. Ensure that `CustomAuth.jar` file is always placed in the *EPM_ORACLE_HOME*`/common/jlib/11.1.2.0` directory.

    For all the servers and clients to work with custom authentication, the `CustomAuth.jar` file must be present in the following two locations:

    * *MIDDLEWARE_HOME*`/user_projects/domains/WEBLOGIC_DOMAIN/lib`
    * *EPM_ORACLE_HOME*`/common/jlib/11.1.2.0`
- Update user directory settings in Shared Services. See Updating Settings in Shared Services.
- Start Shared Services, followed by other EPM System products.
- Test your implementation. See Testing Your Deployment.

Updating Settings in Shared Services

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 Oracle Hyperion Foundation Services.
2. Access Oracle Hyperion Shared Services Console as System Administrator.
3. Select **Administration**, and then **Configure User Directories**.
4. On 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` before starting the following procedure.

To update security options:

1. Access Shared Services Console as System Administrator.
2. Select **Administration**, and then **Configure User Directories**.
3. Select **Security Options**.
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 the EPM System search order.

To test custom authentication:

1. Ensure that EPM System products are running.
2. Access an EPM System component; for example, Oracle Hyperion Enterprise Performance Management 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.

6 Guidelines for Securing EPM System

Related Topics

- Implementing SSL
- Changing the Admin Password
- Regenerating Encryption Keys
- Changing Database Passwords
- Securing Cookies
- Reducing SSO Token Timeout
- Reviewing Security Reports
- Customizing Authentication System for Strong Authentication
- Disabling EPM Workspace Debugging Utilities
- Changing Default Web Server Error Pages
- Support for Third-Party Software

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 Oracle Enterprise Performance Management System environment, secure all communication channels used by your web applications and user directory connections using SSL. See SSL-Enabling EPM System Components.

Additionally, protect all agent ports, for example, port 6861, which is the Oracle Hyperion Reporting and Analysis agent port, using a firewall. End-users do not need to access EPM System agent ports.

Changing the Admin Password

The default Native Directory admin user account provides access to all Oracle Hyperion Shared Services functions. This password is set when you deploy Oracle Hyperion 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 Security Administration Guide*.

Regenerating Encryption Keys

Use the Oracle Hyperion Shared Services Console to periodically regenerate the following:

- Single sign-on token

> **Caution:**
>
> Taskflows used by Oracle Hyperion Financial Management 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

See Regenerating Encryption Keys.

> **Note:**
>
> Oracle Hyperion Shared Services and the security subsystem of Oracle Enterprise Performance Management System use AES encryption with 128-bit key strength.

Changing Database Passwords

Periodically change the password for all Oracle Enterprise Performance Management System product databases. The procedure for changing the database password in Oracle Hyperion 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.

   Oracle Hyperion Shared Services Only:

   > **Note:**
   >
   > In distributed environments where EPM System products are on machines different than Shared Services, you must perform this procedure on all servers.

   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 the 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:

> **Note:**
>
> Follow these steps for the EPM System products deployed on the current server only.

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

4. Start EPM System products and services.

Securing Cookies

Oracle Enterprise Performance Management System web application set 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 Oracle WebLogic Server session descriptor to secure WebLogic Server cookies. Set the value of `cookieSecure` attribute in the `session-param` element to `true`. See Securing Web Applications in Oracle Fusion Middleware Programming Security for Oracle WebLogic Server 11g.

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 Security Administration 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 Oracle Hyperion Shared Services Console on a regular basis, especially to identify failed login attempts across Oracle Enterprise Performance Management 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 Security Administration 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 Using a Custom Authentication Module.

Disabling EPM Workspace Debugging Utilities

- For troubleshooting purposes, Oracle Hyperion Enterprise Performance Management 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 EPM Workspace as administrator.
  2. Select **Navigate**, then **Administer**, and then **Workspace Server Settings**.
  3. In **ClientDebugEnabled** in Workspace Server Settings, select **No**.
  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 to the currently supported JRE version.

A

Custom Authentication Sample Code

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:

```
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;
      }
}
```

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 Oracle Enterprise Performance Management System component.
- 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 component.

Sample Code 2

The following sample code demonstrates the 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.

```
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.)

```
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
```

B
Implementing a Custom Login Class

Oracle Enterprise Performance Management System provides `com.hyperion.css.sso.agent.X509CertificateSecurityAgentImpl` to extract the user identity (`DN`) from x509 certificates.

If you must derive the 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 `com.hyperion.css.sso.agent.X509CertificateSecurityAgentImpl`. Generally, you should customize the `parseCertificate(String sCertificate)` method of this implementation to derive the user name from a certificate attribute other than DN:

```
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 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 nane 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
     * java API can parse the certificate.
     */
    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:

1. 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.

   You can use any name for your custom class.

2. Package the custom login class into `CustomAuth.jar`
3. Copy `CustomAuth.jar` into the deployment:

- **WebLogic:** Copy `CustomAuth.jar` into *`MIDDLEWARE_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 *`MIDDLEWARE_HOME`*`/user_projects/domains/`*`WEBLOGIC_DOMAIN`*`/lib`.

- **Client deployments**: Copy `CustomAuth.jar` into all Oracle Enterprise Performance Management System client deployments, into the following location:

  EPM_ORACLE_HOME`/common/jlib/11.1.2.0`, typically, `Oracle/Middleware/common/jlib/11.1.2.0`

Oracle recommends that you enable Client Certificate Authentication if you are using a custom login class.

C

Migrating Users and Groups Across User Directories

Overview

Many scenarios may cause the user and group identities of provisioned Oracle Enterprise Performance Management 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.
- This Migration procedure does not support the migration of user accounts from a source user directory to a target user directory but only their association in EPM applications. Users have to be created manually in the target user directory. This process is applicable to users of any source user directory, including Native Directory.

  If a source user directory configured with Hyperion Shared Services had groups except Native Directory groups, then those groups should also be created within the target user directory.

Prerequisites

- Oracle Enterprise Performance Management 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

Export Native Directory Data

Follow these steps in the source environment:

Use Oracle Hyperion Enterprise Performance Management System Lifecycle Management to export only the following Shared Services artifacts 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`, who performed the export operation and *`EXPORT_DIR`* is the name of the export directory. Typically, these files are created:

- `Groups.csv`
- `Assigned Roles.csv`
- `Delegated Lists.csv`
- `Assigned Roles/`*`PROD_NAME`*`.csv` for each deployed application, where *`PROD_NAME`* is the name of an Oracle Enterprise Performance Management 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.
> - Ensure that `Users.csv` file is not exported.

After exporting the artifacts, verify that the Migration Status Report displays the status of the last export operation as `Completed`.

To export Native Directory data:

1. In the View pane of Oracle Hyperion Shared Services Console, in the **Foundation** application group, select **Shared Services** application.
2. To migrate, select only the required artifacts from the list below:
   - Native Directory Groups
   - Assigned Roles
   - Delegated Lists
3. Click **Export**.
4. Enter a name for the export archive. Default is `admin` *DATE*; for example `admin 13-03-18`.
5. Click **Export**.

Import Native Directory Data

Follow these steps in the target environment:

1. Manually create:
   a. Users in the target external user directory, similar to source user directory.
   b. Groups in the target external user directory, similar to source user directory, except Native Directory groups.
2. Configure the Target 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 Security Administration 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 Oracle Hyperion 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 Security Administration 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.
3. Move the Target User Directory to top of the Search Order.

   > **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 order enables 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 Security Administration Guide*.
4. Restart Oracle Hyperion Foundation Services and other EPM System components to enforce the changes that you made.
5. Import Native Directory Data (exported from the source environment):
   Run Lifecycle Management with `create/update` option to import the data that you exported earlier (as listed below) from Native Directory.

- `Groups.csv`
- `Assigned Roles.csv`
- `Delegated Lists.csv`

> **Note:**
>
> - See the *Oracle Enterprise Performance Management System Lifecycle Management Guide* for detailed instructions on importing data using Lifecycle Management.
> - Ensure that `Users.csv` file is not imported.

After importing data, verify that the Migration Status Report displays the status of the last import operation as `Completed`.

To import Native Directory data:

a. In the View pane of Shared Services Console, expand **File System**.

b. Select the file system location of the import files.

c. Select the type of artifacts for which you want to import provisioning information.

d. Click **Import**.

e. Click **OK**.

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

Oracle Hyperion 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 Planning when assigning access to data forms, members, and task lists.

Financial Management

Oracle Hyperion 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 Financial Management repository with that in Native Directory.