Oracle® Hyperion Enterprise Performance Management System

Security Administration Guide

RELEASE 11.1.2.1

ORACLE

ENTERPRISE PERFORMANCE
MANAGEMENT SYSTEM

EPM System Security Administration Guide, 11.1.2.1

Authors: EPM Information Development Team

Contents

Documentation Accessibility

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

Access to Oracle Support

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

1 About EPM System Security

In This Chapter

About EPM System Products

Oracle Hyperion 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's Hyperion® Foundation Services
- Oracle Essbase
- Oracle Hyperion Financial Management, Fusion Edition
- Oracle Hyperion Planning, Fusion Edition
- Oracle's Hyperion Reporting and Analysis

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

Assumed Knowledge

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

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

    - 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

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

- An optional access management system; for example, Oracle Access Manager to provide SSO access to EPM System products
- Use of an integrated SSO infrastructure; for example, Kerberos.

  You can use Kerberos authentication with the access management system (SiteMinder) to ensure that Windows users can transparently log into SiteMinder and EPM System products.
- Use of Secure Socket Layer (SSL) to secure communication channels among EPM System products and clients

User Authentication

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

Authentication Components

The following sections describe the components that support SSO:

- "Native Directory" on page 12
- "External User Directories" on page 13

Native Directory

Native Directory refers to the relational database that Oracle's 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's Hyperion® Shared Services Console. See "Managing Native Directory" in the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

External User Directories

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

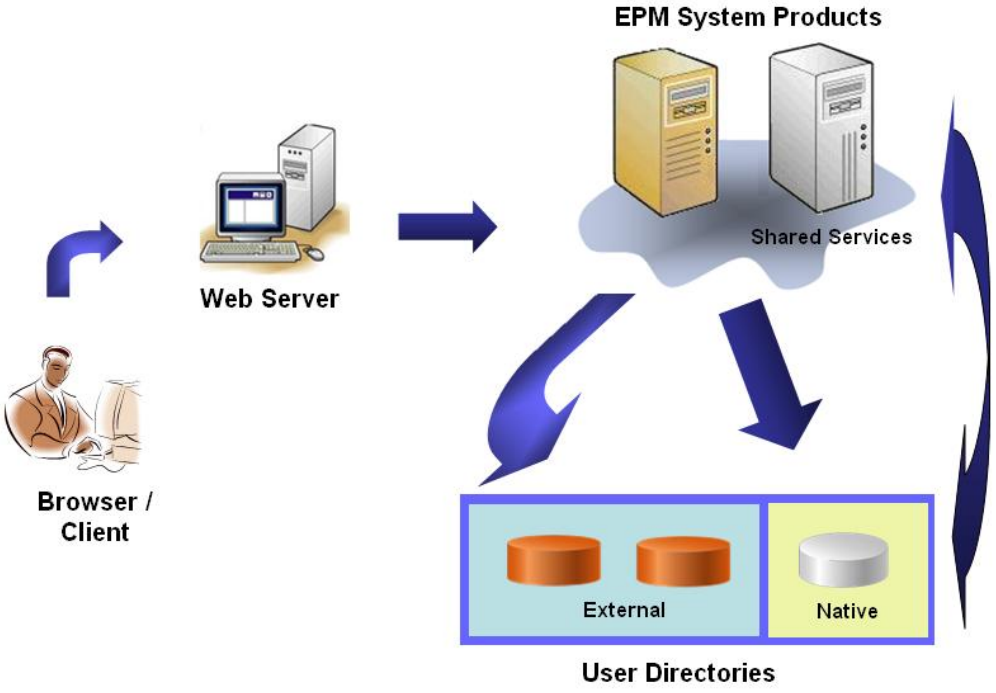
EPM System products are supported on several user directories, including LDAP-based user directories, such as Oracle Internet Directory, Sun Java System Directory Server (formerly SunONE Directory Server), and Microsoft Active Directory. Relational databases and SAP native repository also are supported as user directories. User directories other than Native Directory are referred to as external user directories throughout this document.

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

Default EPM System Single Sign-on

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

The default SSO process is depicted in the following illustration.



1. Through a browser, users access a EPM System product login screen and enter user names and passwords.

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

   Access is denied if a user account is not found in any configured user directory.
2. Using the retrieved user information, the EPM System product queries Native Directory to obtain provisioning details for the user.
3. EPM System product checks the Access Control List (ACL) in the product to determine the application artifacts that the user can access.

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

Single Sign-on with SAP

EPM System can be deployed to support SSO with SAP Enterprise Portal where users who are defined in an SAP native repository are permitted to navigate between the SAP Portal and EPM System products by accepting an SAP logon ticket. In this scenario, EPM System treats SAP Enterprise Portal like a application. If SAP BW or R/3 native repository is configured as an external directory in Shared Services, users can log in to EPM System products using the user ID and password stored in the SAP system.

See "Single Sign-on with SAP Enterprise Portal" on page 113.

Single Sign-on from Access Management Systems

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

SSO from security agents is available for EPM System web applications only. In this scenario, EPM System products use the user information provided by the security agent to determine access permissions of users. To enhance security, Oracle recommends that direct access to the servers be blocked by firewalls so that all requests are routed through an SSO portal.

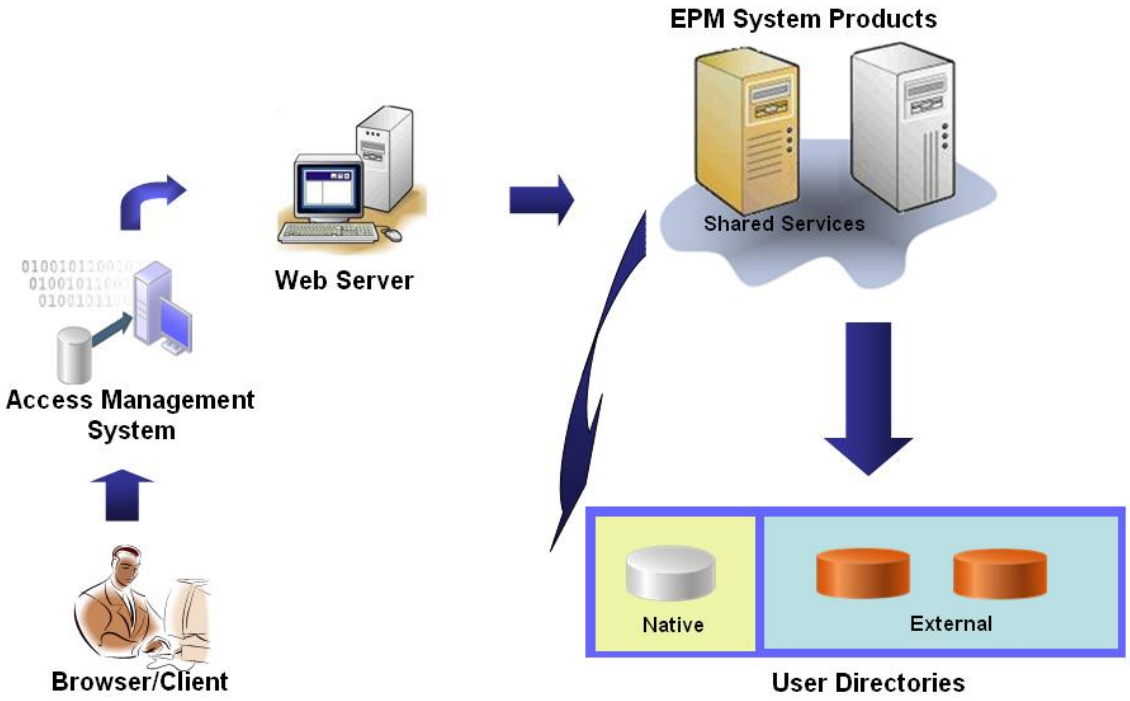
SSO from access management systems is supported by accepting authenticated user credentials through an acceptable SSO mechanism. See "Supported SSO Methods" on page 85. The access management system authenticates users and passes their login name to EPM System. EPM System verifies the login names against configured user directories.

See these topics.

- "Single Sign-on from Oracle Access Manager" on page 87
- "OracleAS Single Sign-on" on page 88

- "SiteMinder SSO" on page 98
- "Kerberos Single Sign-on" on page 102
- "Single Sign-on with SAP Enterprise Portal" on page 113

The illustrated concept:



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

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

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

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

   If the user logged on to SAP Portal, an SAP logon ticket is passed to the EPM System product, which decrypts the SAP logon ticket using an SAP certificate.

   The access management system passes the user's login name (value of `Login Attribute`) to the EPM System product using an acceptable SSO mechanism. See "Supported SSO Methods" on page 85.

2. To verify user credentials, the EPM System product tries to locate the user in a user directory. If a matching user account is found, user information is returned to the EPM System

product. EPM System security sets the SSO token that enables SSO across EPM System products.

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

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

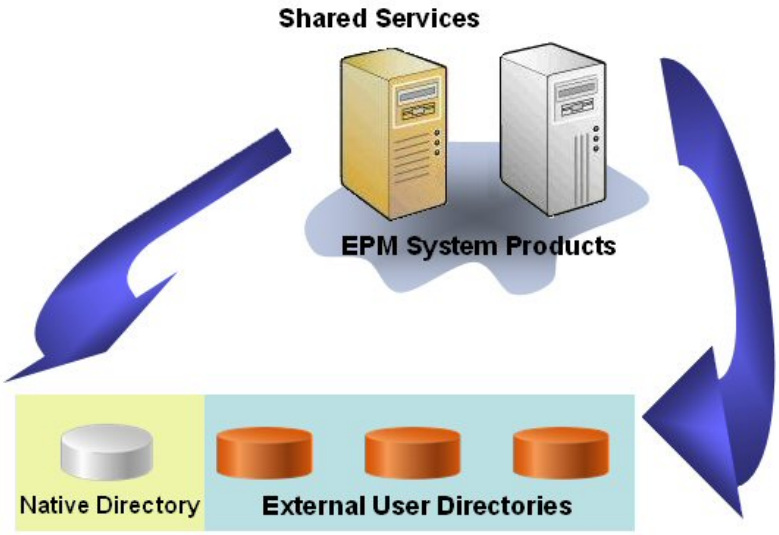
Provisioning (Role-Based Authorization)

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

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

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

An illustrated overview of the authorization process:



1. After a user is authenticated, EPM System product queries user directories to determine the user's groups.

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

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

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

Roles

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

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

Global Roles

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

Predefined Roles

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

Aggregated Roles

Aggregated roles, also known as custom roles, aggregate multiple predefined roles belonging to an application. An aggregated role can contain other aggregated roles. For example, a Shared Services Administrator or Provisioning Manager can create an aggregated role that combines the Planner and View User roles of a Planning application. Aggregating roles can simplify the administration of applications that 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.

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.

2 SSL-Enabling EPM System Components

In This Chapter

Assumptions

- You have determined the deployment topology and identified the communication links that are to be secured using SSL.
- You have obtained the required certificates from a Certificate Authority (CA), either a well-known CA or your own, or created self-signed certificates. See "Required Certificates" on page 31.
- You are familiar with SSL concepts and procedures such as importing certificates.

  See "Information Sources" on page 19 for a list of reference documents.

Information Sources

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

- **Oracle WebLogic Server**: See "Configuring SSL" in the *Securing WebLogic Server Guide*.
- **Oracle HTTP Server:**: See the following topics in the *Oracle HTTP Server Administrator's Guide*:
  - Managing Security
  - Enabling SSL for Oracle HTTP Server
- **User Directories**: See the documentation from the user directory vendor. Useful links:
  - **Oracle Internet Directory:** See Oracle Internet Directory Administrator's Guide
  - **Sun Java System Directory Server:** See "Directory Server Security" in the *Sun Java System Directory Server Administration Guide*
  - **Active Directory:** See these documents:
    - Microsoft Windows Server 2008 Active Directory documentation
    - Microsoft Windows Server 2003 Active Directory documentation
  - **Novell eDirectory:** Novell eDirectory documentation
- **Databases**: See the documentation from the database vendor.
- **Internet Information Services**: See How to Implement SSL in IIS.

References

This document refers to the following installation and deployment locations:

- *MIDDLEWARE_HOME* refers to the location of middleware components such as WebLogic Server, and, optionally, one or more *EPM_ORACLE_HOME*. The *MIDDLEWARE_HOME* is defined during EPM System product installation. The default *MIDDLEWARE_HOME* directory is `Oracle/Middleware`.
- *EPM_ORACLE_HOME* refers to the installation directory containing the files required to support EPM System products. *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 Oracle's EPM System Products

The EPM System deployment process automatically deploys Oracle's EPM System products to work in both SSL and non-SSL modes. For example, a default deployment of Shared Services identifies port 28080 as the HTTP port and 28443 as the secure port. See *Oracle Hyperion Enterprise Performance Management System Installation Start Here* for a list of ports that EPM System uses.

While deploying Shared Services, you specify whether to use SSL for the entire EPM System deployment.

Caution! If you select SSL mode for one product, it is automatically selected for all products.

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

Supported SSL Scenarios

The following SSL scenarios are supported:

- Full SSL deployment. See "Full SSL Deployment of EPM System" on page 32.
- SSL termination at the web Server. See "Terminating SSL at the Web Server" on page 53.
- SSL deployment with an offloader. See "Deploying EPM System with an SSL Offloader" on page 59.
- Two-way SSL deployment. See "Client Certificate Authentication (Two-Way SSL)" on page 67.

Firewall Considerations

You use firewalls to enhance security by denying unauthorized access to resources. Depending on the location of the firewall, you must open firewall ports to permit communication among EPM System components. For example, if you use a firewall between Oracle HTTP Server and clients (browsers), you must ensure that the port that Oracle HTTP Server uses for external communication is open on the firewall. Similarly, if you deploy a firewall between Oracle HTTP Server and WebLogic Server, ensure that the ports that EPM System components use for server-to-server communications are open on the firewall.

See *Oracle Hyperion Enterprise Performance Management System Installation Start Here* for a list of ports that EPM System components use.

Securable EPM System Connections

Subtopics

Essbase Domain

Table 1 Essbase Domain Connections that can be SSL-Enabled

Source	Destination	Protocol	Server-to-Server
Browser	Oracle Essbase Administration Services web application	HTTP/HTTPS	No
Browser	Shared Services web application	HTTP/HTTPS	No
Oracle Hyperion Smart View for Office, Fusion Edition	Shared Services web application	HTTP/HTTPS	No
Smart View	Oracle Hyperion Provider Services web application	HTTP/HTTPS	No
Administration Services (Desktop)	Administration Services web application	HTTP/HTTPS	No
Essbase Excel Add-in	Oracle Essbase Studio Server	HTTP/HTTPS	No
Administration Services web application	Shared Services web application	HTTP/HTTPS	Yes
Essbase Server	Shared Services web application	HTTP/HTTPS	Yes
Shared Services web application	Corporate User Directories	LDAP/LDAPS	Yes
Provider Services web application	Corporate User Directories	LDAP/LDAPS	Yes
Essbase Server	Corporate User Directories	LDAP/LDAPS	Yes
Provider Services web application	Relational Database	JDBC/JDBC over SSL	Yes
Essbase Server	Relational Database	JDBC/JDBC over SSL	Yes
Shared Services web application	Relational Database	JDBC/JDBC over SSL	Yes

Planning Domain

Table 2 Planning Domain Connections that can be SSL-Enabled

Source	Destination	Protocol	Server-to-Server
Browser	Hyperion Calculation Manager web application	HTTP/HTTPS	No
Browser	Administration Services web application	HTTP/HTTPS	No
Browser	Shared Services web application	HTTP/HTTPS	No
Browser	Oracle Hyperion Financial Reporting, Fusion Edition, web application	HTTP/HTTPS	No
Browser	Planning web application	HTTP/HTTPS	No
Browser	Reporting and Analysis web application	HTTP/HTTPS	No
Browser	Oracle's Hyperion® Web Analysis web Application	HTTP/HTTPS	No
Browser	Oracle Enterprise Performance Management Workspace, Fusion Edition, web application	HTTP/HTTPS	No
Administration Services (Desktop)	Administration Services web application	HTTP/HTTPS	No
Financial Reporting Studio	Provider Services web application	HTTP/HTTPS	No
Financial Reporting Studio	Financial Reporting web application	HTTP/HTTPS	No
Financial Reporting Studio	EPM Workspace web application	HTTP/HTTPS	No
Smart View	Shared Services web application	HTTP/HTTPS	No
Smart View	Provider Services web application	HTTP/HTTPS	No
Smart View	Financial Reporting web application	HTTP/HTTPS	No
Smart View	Planning web application	HTTP/HTTPS	No
Smart View	Web Analysis web application	HTTP/HTTPS	No
Smart View	EPM Workspace web application	HTTP/HTTPS	No
Planning web application	Administration Services web application	HTTP/HTTPS	Yes
Planning web application	Shared Services web application	HTTP/HTTPS	Yes
Financial Reporting web application	Provider Services web application	HTTP/HTTPS	Yes
Financial Reporting web application	EPM Workspace web application	HTTP/HTTPS	Yes
Shared Services web application	Planning web application	HTTP/HTTPS	Yes
Calculation Manager web application	Shared Services web application	HTTP/HTTPS	Yes
Web Analysis web application	Shared Services web application	HTTP/HTTPS	Yes

Source	Destination	Protocol	Server-to-Server
Administration Services web application	Shared Services web application	HTTP/HTTPS	Yes
Essbase Server	Shared Services web application	HTTP/HTTPS	Yes
Provider Services web application	Planning web application	HTTP/HTTPS	Yes
Financial Reporting web application	Provider Services web application	HTTP/HTTPS	Yes
Reporting and Analysis Services	Relational Database	JDBC/JDBC over SSL	Yes
Calculation Manager web application	Relational Database	JDBC/JDBC over SSL	Yes
Shared Services web application	Relational Database	JDBC/JDBC over SSL	Yes
Administration Services web application	Relational Database	JDBC/JDBC over SSL	Yes
Web Analysis web application	Relational Database	JDBC/JDBC over SSL	Yes
Financial Reporting web application	Relational Database	JDBC/JDBC over SSL	Yes
EPM Workspace web application	Relational Database	JDBC/JDBC over SSL	Yes
Planning web application	Corporate User Directories	LDAP/LDAPS	Yes
Provider Services web application	Corporate User Directories	LDAP/LDAPS	Yes
Essbase Server	Corporate User Directories	LDAP/LDAPS	Yes

Financial Management Domain

Table 3 Financial Management Domain Connections that can be SSL-Enabled

Source	Destination	Protocol	Server-to-Server
Browser	Shared Services web application	HTTP/HTTPS	No
Browser	Calculation Manager web application	HTTP/HTTPS	No
Browser	Financial Reporting web application	HTTP/HTTPS	No
Browser	Reporting and Analysis web application	HTTP/HTTPS	No
Browser	Web Analysis web application	HTTP/HTTPS	No
Browser	EPM Workspace web application	HTTP/HTTPS	No
Browser	Financial Management web application	HTTP/HTTPS	No
Financial Management Console	Shared Services web application	HTTP/HTTPS	No
Financial Reporting Studio	Financial Reporting web application	HTTP/HTTPS	No
Financial Reporting Studio	EPM Workspace web application	HTTP/HTTPS	No

Source	Destination	Protocol	Server-to-Server
Smart View	Shared Services web application	HTTP/HTTPS	No
Smart View	Financial Reporting web application	HTTP/HTTPS	No
Smart View	Web Analysis web application	HTTP/HTTPS	No
Smart View	EPM Workspace web application	HTTP/HTTPS	No
Smart View	Financial Management Smart View web application	HTTP/HTTPS	No
Shared Services web application	Oracle Hyperion Enterprise Performance Management System Lifecycle Management web application for Financial Management	HTTP/HTTPS	Yes
Reporting and Analysis Services	Corporate User Directories	LDAP/LDAPS	Yes
Shared Services web application	Corporate User Directories	LDAP/LDAPS	Yes
Financial Reporting web application	Shared Services web application	HTTP/HTTPS	Yes
Financial Reporting web application	EPM Workspace web application	HTTP/HTTPS	Yes
Financial Management Server	Shared Services web application	HTTP/HTTPS	Yes
Web Analysis web application	Shared Services web application	HTTP/HTTPS	Yes
Shared Services web application	Relational Database	JDBC/JDBC over SSL	Yes
Web Analysis web application	Relational Database	JDBC/JDBC over SSL	Yes
EPM Workspace web application	Relational Database	JDBC/JDBC over SSL	Yes

Reporting and Analysis Domain

Table 4 Reporting and Analysis Domain Connections that can be SSL-Enabled

Source	Destination	Protocol	Server-to-Server
Browser	Shared Services web application	HTTP/HTTPS	No
Browser	Financial Reporting web application	HTTP/HTTPS	No
Browser	Reporting and Analysis web application	HTTP/HTTPS	No
Browser	Web Analysis web application	HTTP/HTTPS	No
Browser	EPM Workspace web application	HTTP/HTTPS	No
Financial Reporting Studio	Provider Services web application	HTTP/HTTPS	No
Financial Reporting Studio	Financial Reporting web application	HTTP/HTTPS	No
Financial Reporting Studio	EPM Workspace web application	HTTP/HTTPS	No

Source	Destination	Protocol	Server-to-Server
Smart View	Shared Services web application	HTTP/HTTPS	No
Smart View	Financial Reporting web application	HTTP/HTTPS	No
Smart View	Web Analysis web application	HTTP/HTTPS	No
Smart View	EPM Workspace web application	HTTP/HTTPS	No
Financial Reporting web application	Shared Services web application	HTTP/HTTPS	Yes
Financial Reporting web application	EPM Workspace web application	HTTP/HTTPS	Yes
Financial Reporting web application	Provider Services web application	HTTP/HTTPS	Yes
Web Analysis web application	Provider Services web application	HTTP/HTTPS	Yes
Web Analysis web application	Shared Services web application	HTTP/HTTPS	Yes
Web Analysis web application	Corporate User Directories	LDAP/LDAPS	Yes
Shared Services web application	Corporate User Directories	LDAP/LDAPS	Yes
Oracle's Hyperion® Interactive Reporting Studio	Relational Database	JDBC/JDBC over SSL	No
Shared Services web application	Relational Database	JDBC/JDBC over SSL	Yes
Web Analysis web application	Relational Database	JDBC/JDBC over SSL	Yes
EPM Workspace web application	Relational Database	JDBC/JDBC over SSL	Yes

Profitability and Cost Management Domain

Table 5 Profitability and Cost Management Domain Connections that can be SSL-Enabled

Source	Destination	Protocol	Server-to-Server
Browser	Administration Services web application	HTTP/HTTPS	No
Browser	Shared Services web application	HTTP/HTTPS	No
Browser	Financial Reporting web application	HTTP/HTTPS	No
Browser	Reporting and Analysis web application	HTTP/HTTPS	No
Browser	Web Analysis web application	HTTP/HTTPS	No
Browser	EPM Workspace web application	HTTP/HTTPS	No
Browser	Oracle Hyperion Profitability and Cost Management, Fusion Edition, web application	HTTP/HTTPS	No
Administration Services (Desktop)	Administration Services web application	HTTP/HTTPS	No

Source	Destination	Protocol	Server-to-Server
Financial Reporting Studio	Financial Reporting web application	HTTP/HTTPS	No
Financial Reporting Studio	EPM Workspace web application	HTTP/HTTPS	No
Smart View	Provider Services web application	HTTP/HTTPS	No
Smart View	Shared Services web application	HTTP/HTTPS	No
Smart View	Financial Reporting web application	HTTP/HTTPS	No
Smart View	Web Analysis web application	HTTP/HTTPS	No
Smart View	EPM Workspace web application	HTTP/HTTPS	No
Shared Services web application	Profitability and Cost Management web application	HTTP/HTTPS	Yes
Administration Services web application	Shared Services web application	HTTP/HTTPS	Yes
Financial Reporting web application	Shared Services web application	HTTP/HTTPS	Yes
Financial Reporting web application	Provider Services web application	HTTP/HTTPS	Yes
Financial Reporting web application	EPM Workspace web application	HTTP/HTTPS	Yes
Web Analysis web application	Provider Services web application	HTTP/HTTPS	Yes
Web Analysis web application	Shared Services web application	HTTP/HTTPS	Yes
Profitability and Cost Management web application	Shared Services web application	HTTP/HTTPS	Yes
Profitability and Cost Management web application	Corporate User Directories	LDAP/LDAPS	Yes
Administration Services web application	Corporate User Directories	LDAP/LDAPS	Yes
Provider Services web application	Corporate User Directories	LDAP/LDAPS	Yes
Shared Services web application	Corporate User Directories	LDAP/LDAPS	Yes
Profitability and Cost Management web application	Relational Database	JDBC/JDBC over SSL	Yes
Administration Services web application	Relational Database	JDBC/JDBC over SSL	Yes
Shared Services web application	Relational Database	JDBC/JDBC over SSL	Yes
Web Analysis web application	Relational Database	JDBC/JDBC over SSL	Yes
Provider Services web application	Relational Database	JDBC/JDBC over SSL	Yes
EPM Workspace web application	Relational Database	JDBC/JDBC over SSL	Yes

FDM Domain

Table 6 FDM Domain Connections that can be SSL-Enabled

Source	Destination	Protocol	Server-to-Server
Browser	Shared Services web application	HTTP/HTTPS	No
Browser	EPM Workspace web application	HTTP/HTTPS	No
Browser	Oracle Hyperion Financial Data Quality Management ERP Integration Adapter for Oracle Applications Adapter web application	HTTP/HTTPS	No
Browser	Oracle Hyperion Financial Data Quality Management, Fusion Edition, web application	HTTP/HTTPS	No
FDM Server	Shared Services web application	HTTP/HTTPS	Yes
ERP Integrator Adapter web application	Shared Services web application	HTTP/HTTPS	Yes
ERP Integrator Adapter web application	Corporate User Directories	LDAP/LDAPS	Yes
FDM Server	Corporate User Directories	LDAP/LDAPS	Yes
Shared Services web application	Corporate User Directories	LDAP/LDAPS	Yes
Shared Services web application	Relational Database	JDBC/JDBC over SSL	Yes
ERP Integrator Adapter web application	Relational Database	JDBC/JDBC over SSL	Yes
FDM Server	Relational Database	JDBC/JDBC over SSL	Yes
EPM Workspace web application	Relational Database	JDBC/JDBC over SSL	Yes

Performance Management Architect Domain

Table 7 Performance Management Architect Domain Connections that can be SSL-Enabled

Source	Destination	Protocol	Server-to-Server
Oracle Hyperion EPM Architect, Fusion Edition, Batch Client	EPM Workspace web application	HTTP/HTTPS	No
Performance Management Architect Generator	Performance Management Architect web application	HTTP/HTTPS	No
Performance Management Architect Generator	Financial Management web application	HTTP/HTTPS	No
Performance Management Architect Generator	Planning web application	HTTP/HTTPS	No
Performance Management Architect Generator	EPM Workspace web application	HTTP/HTTPS	No

Source	Destination	Protocol	Server-to-Server
Essbase Studio	Performance Management Architect Dimension Server	HTTP/HTTPS	Yes
Calculation Manager web application	Performance Management Architect Dimension Server	HTTP/HTTPS	Yes
Calculation Manager web application	Performance Management Architect web application	HTTP/HTTPS	Yes
Data Synchronization web application	Performance Management Architect Dimension Server	HTTP/HTTPS	Yes
Data Synchronization web application	Performance Management Architect web application	HTTP/HTTPS	Yes
Data Synchronization web application	Planning web application	HTTP/HTTPS	Yes
Performance Management Architect NET JNI Bridge	Shared Services web application	HTTP/HTTPS	Yes
Performance Management Architect web application	Data Synchronization web application	HTTP/HTTPS	Yes
Performance Management Architect web application	Performance Management Architect Dimension Server	HTTP/HTTPS	Yes
Performance Management Architect web application	Essbase Studio Server	HTTP/HTTPS	Yes
Performance Management Architect web application	Financial Management web application	HTTP/HTTPS	Yes
Performance Management Architect web application	Planning web application	HTTP/HTTPS	Yes
Performance Management Architect web application	Shared Services web application	HTTP/HTTPS	Yes
Financial Management web application	Performance Management Architect web application	HTTP/HTTPS	Yes
Planning web application	Performance Management Architect web application	HTTP/HTTPS	Yes
Profitability and Cost Management web application	Performance Management Architect Dimension Server	HTTP/HTTPS	Yes
Profitability and Cost Management web application	Performance Management Architect web application	HTTP/HTTPS	Yes
Shared Services web application	Performance Management Architect web application	HTTP/HTTPS	Yes
Data Synchronization web application	Corporate User Directories	LDAP/LDAPS	Yes
Performance Management Architect NET JNI Bridge	Corporate User Directories	LDAP/LDAPS	Yes

Source	Destination	Protocol	Server-to-Server
Performance Management Architect web application	Corporate User Directories	LDAP/LDAPS	Yes
Data Synchronization web application	Relational Database	JDBC/JDBC over SSL	Yes
Performance Management Architect NET JNI Bridge	Relational Database	JDBC/JDBC over SSL	Yes
Performance Management Architect Dimension Server	Relational Database	JDBC/JDBC over SSL	Yes

Financial Close Management Domain

Table 8 Financial Close Management Domain Connections that can be SSL-Enabled

Source	Destination	Protocol	Server-to-Server
Browser	EPM Workspace web application	HTTP/HTTPS	No
Browser	EPM Workspace web application	HTTP/HTTPS	No
Smart View	EPM Workspace web application	HTTP/HTTPS	No
Oracle Hyperion Financial Close Management web application	Shared Services web application	HTTP/HTTPS	Yes
Financial Close Management web application	SOA Server	HTTP/HTTPS	Yes
Financial Close Management web application	Financial Management Web Services web application	HTTP/HTTPS	Yes
SOA Server	Financial Management Web Services web application	HTTP/HTTPS	Yes
SOA Server	Financial Close Management web application	HTTP/HTTPS	Yes
Financial Close Management web application	Corporate User Directories	LDAP/LDAPS	Yes
Financial Close Management web application	Relational Database	JDBC/JDBC over SSL	Yes

Disclosure Management Domain

Table 9 Disclosure Management Domain Connections that can be SSL-Enabled

Source	Destination	Protocol	Server-to-Server
Browser	EPM Workspace web application	HTTP/HTTPS	No
Browser	Oracle Hyperion Disclosure Management web application	HTTP/HTTPS	No

Source	Destination	Protocol	Server-to-Server
Smart View	Disclosure Management web application	HTTP/HTTPS	No
Smart View	EPM Workspace web application	HTTP/HTTPS	No
Disclosure Management web application	Shared Services web application	HTTP/HTTPS	Yes
Disclosure Management web application	Corporate User Directories	LDAP/LDAPS	Yes
Disclosure Management web application	Relational Database	JDBC/JDBC over SSL	Yes

Required Certificates

SSL communication uses certificates to establish a trust between components. Oracle recommends that you use certificates from well-known third-party CAs to SSL-enable EPM System in a production environment. You may use self-signed certificates for test purposes if a root CA is not available to sign certificates.

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 between Provider Services web application and other EPM System server components, you must disable host name verification for Provider Services web application in WebLogic Server.

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

- A root CA certificate.

  Note: You do not need to install 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 pre-loaded with certificates of well-known third-party CAs. If you are acting as your own CA, you must import your CA root certificate into the keystore used by the clients accessed from such browsers. For example; if you are acting as your own CA, Web Analysis clients cannot establish an SSL handshake with the server if your CA root certificate is not available to the browser from which Web Analysis is accessed.

- Signed certificates for each machine in your deployment.

  Note: Full SSL scenario requires two signed certificates for Oracle HTTP Server, one for internal communication and the other for external communication.

  Note: In scenarios where the client must present SSL

Installing EPM System

The SSL configuration process described in this document assumes that the required EPM System components are installed on each server. See the following documents for detailed information:

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

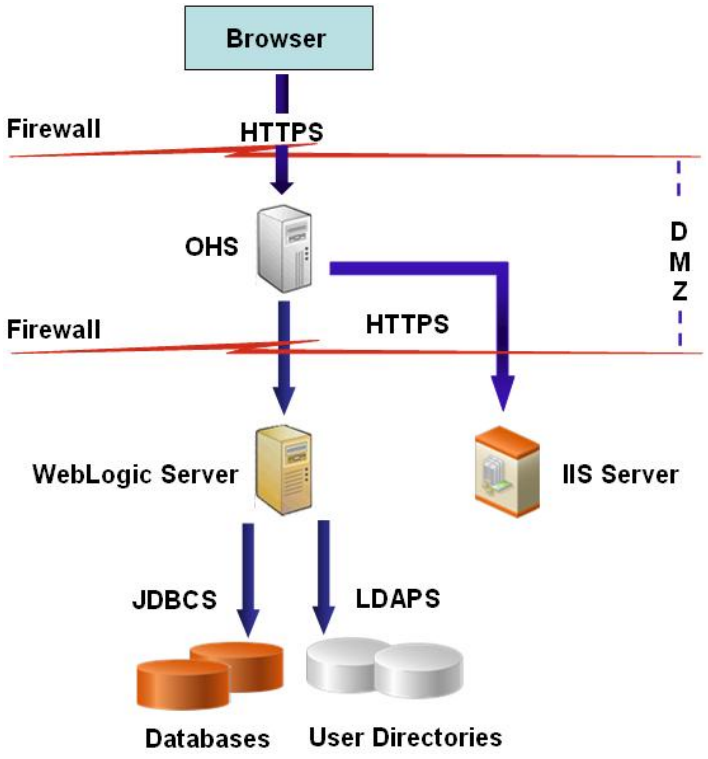
Full SSL Deployment of EPM System

Subtopics

Deployment Architecture

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

The illustrated concept:



Note: Oracle HTTP Server uses `mod_wl_ohs` for redirection to WebLogic Server and mod_proxy for redirection to IIS.

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

Assumptions

- EPM System components are installed on the server but are not configured. In addition to the EPM System components you select, the default EPM System installation process installs WebLogic Server and Oracle HTTP Server within *MIDDLEWARE_HOME*.
- Your deployment architecture comprises the following servers:
  - A web server running Oracle HTTP Server, which proxies HTTP requests to the application servers.
  - Computers that host WebLogic Server, which runs EPM System components such as Planning and Foundation Services.
  - One or more computers that host Internet Information Services (IIS), which runs the following components:
    - Performance Management Architect
    - Financial Management
    - FDM
    - Oracle Hyperion Strategic Finance, Fusion Edition

    The EPM System expects the IIS Server to be SSL-enabled. See Microsoft documentation "How to implement SSL in IIS" for instructions.
  - A server that hosts a database; for example, Oracle Database.
  - A server that hosts a supported user directory, such as Sun Java System Directory Server or Microsoft Active Directory.
- You are using signed certificates from a trusted third-party CA.

  Note: If you are using your own CA to sign certificates, Oracle assumes that you are well-versed in generating certificate requests and signing your own certificates.

Process Overview

Configuring SSL includes the following steps:

1. Install EPM System. See "Installing EPM System" on page 32.
2. SSL-enable the database and database client. See your database documentation for detailed procedures.

3. Add server aliases on the web server. See "Adding Server Aliases" on page 37.
4. If you are not using certificates from a well-known third-party CA, prepare Oracle's Hyperion Enterprise Performance Management System Configurator by importing the required root certificates. You must import root CA certificate into each server in which EPM System components are installed. See "Preparing EPM System Configurator" on page 39.

   Import the root certificate of the CA that signed the following certificates into the EPM System Configurator keystore (*MIDDLEWARE_HOME*/jdk160_11/jre/lib/security/cacerts):

   - Database server used for Shared Services Registry and Shared Services database
   - Database server used by EPM System components
   - SMTP mail server
   - User directory servers
   - Application servers
   - Web servers
5. Using EPM System Configurator, configure and deploy EPM System components. You must select the required SSL settings before deploying the web components to the application server. See "Configuring and Deploying EPM System" on page 40.
6. Configure EPM System web components for SSL communication. See "Configuring EPM System Web Components Deployed on WebLogic Server" on page 46.
7. Configure the web server (Oracle HTTP Server) for SSL communication. See "Configuring the Web Server" on page 44.
8. Configure Oracle HTTP Server for redirection to IIS. (Redirection from Oracle HTTP Server to IIS is automatically performed by EPM System Configurator.) See "Optional: Setting up Redirection from Oracle HTTP Server Load Balancer to Internet Information Services" on page 48.

Configuring EPM System for Full SSL

Subtopics

SSL-enabling the Database Server and Client

You must SSL-enable the database server and client before EPM System Configurator can establish an OLEDB, JDBC, or ODBC SSL connection to the database.

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

If you are using Oracle database, Oracle Data Access Components for Oracle Client must be installed on the machines that host the following EPM System components:

- Performance Management Architect (a Foundation Services component)
- Financial Management
- Strategic Finance

See Oracle Data Access Components for Oracle Client documentation for instructions on SSL-related tasks.

Creating a Custom Keystore on WebLogic Server and Importing Certificates

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

This keystore will be used for managing incoming SSL requests to WebLogic Server.

Note: Perform this procedure on each WebLogic Server machine to which EPM System components will be deployed.

➤ To create a custom keystore and import certificate:

1 From a console, change directory to *MIDDLEWARE_HOME*/jdk160_11/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 **Optional:** If you are not using a well-known third-party CA to sign the server certificate, execute a keytool command such as the following to import the root CA certificate into the custom keystore or into an available Java keystore:

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

Note: The CA certificate must be available to WebLogic Server. See step 10 in "Configuring EPM System Web Components Deployed on WebLogic Server" on page 46.

6 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
```

Installing root CA certificate for WebLogic Server

A WebLogic Server is installed when you install EPM System components. If you use a different WebLogic Server installation to host EPM System components, perform the following procedure on that installation to import the root CA certificate into the Sun JVM and JRockit JVM keystores of the WebLogic Server.

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

Note: If you are using a CA whose root certificate is available in the keystore, you do not need to perform this procedure.

Note: Perform this procedure on each WebLogic Server machine to which EPM System components will be deployed. This certificate is used to manage outgoing SSL requests from WebLogic Server.

➤ To install root CA certificate for WebLogic Server using keytool:

1 **If you are not using a well-known third-party CA, copy the root CA certificate into a local directory on the machine where WebLogic Server is installed.**

2 **From a console, change directory to** *MIDDLEWARE_HOME*`/jdk160_11/jre/bin`.

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

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

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

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

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

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

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

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

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

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

Adding Server Aliases

Full SSL configuration uses two server aliases, one for external communication between Oracle HTTP Server and browsers, and the other for routing internal communication among EPM System servers. Oracle recommends that both of these certificates be tied to server aliases, for example, `epm.myCompany.com` and `empinternal.myCompany.com`, to prevent the exposure of server names.

You add server aliases in the `hosts` file on the machine. Ensure that the server aliases point to the IP address of the machine that hosts Oracle HTTP Server. Clients must resolve these aliases through DNS.

Creating Wallets and Installing Certificates for Oracle HTTP Server

Oracle HTTP Server certificates are stored in Oracle Wallet. You need two wallets, which can be created using Oracle Wallet Manager.

The two signed certificates—one for external communication between Oracle HTTP Server and browsers, and the other for routing internal communication among EPM System servers—are required to support full SSL configuration. Oracle recommends that these certificates be tied to server aliases, for example, `epm.myCompany.com` and `empinternal.myCompany.com`, to prevent the exposure of server names and to enhance security. If you are not using a third-party CA known to Oracle HTTP Server, you also need the root CA certificate.

Note: Perform this procedure on each Oracle HTTP Server host machine.

➤ To install Oracle HTTP Server certificate into Oracle Wallet:

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

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

  `xxxxxx` is the Oracle HTTP Server instance number.

- UNIX: Execute *`MIDDLEWARE_HOME`*`/ohs/bin/owm` to launch the Wallet Manager from the command line.

  Note: The Wallet Manager requires a graphical environment.

2 **Create a new, empty Wallet.**

a. In Oracle Wallet Manager, select **Wallet**, and then **New**.

b. Click **Yes** to create a default wallet directory, or **No** to create the Wallet file in a location of your choice.

c. In **Wallet Password** and **Confirm Password** on the New Wallet screen, enter the password that you want to use.

d. Click **OK**.

e. Click **No** in the confirmation dialog box.

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 screen, enter the required information.

   For 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. Click **OK** in the confirmation dialog box.

e. Right-click the certificate request that you created and 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 **Select Wallet, and then Auto Login to activate auto login.**

8 **Save the Wallet to a convenient location; for example,** *MIDDLEWARE_HOME*`/ohs/bin/wallet/epmwallet`.

9 **Repeat step 2 to step 9 to create another wallet and install certificates.**

Preparing EPM System Configurator

During the configuration process, EPM System Configurator must establish secure communication with the components for which SSL is to be supported. To establish secure communication, you must import the root certificate of the CA that signed the certificate that was used to SSL-enable these components.

- Database server used for Shared Services Registry and Shared Services database
- Database server used by EPM System components
- SMTP mail server
- Application server
- Web server

Generally, the root certificates of well-known third-party CAs are available in the keystore that EPM System Configurator uses, so you do not need to reimport them.

If you did not use signed certificates from a well-known third-party CA (if you used self-signed certificates, for example), you must import the root CA certificate into the keystore that is used by EPM System Configurator. The default EPM System Configurator keystore is *MIDDLEWARE_HOME*`/jdk160_11/jre/lib/security/cacerts`.

You use a tool such as keytool to import certificates into a keystore.

Note: Perform this procedure for each EPM System Configurator that you use to deploy EPM System components.

➤ To import certificates using keytool:

1 **From a console, change directory to** *MIDDLEWARE_HOME*`/jdk160_11/jre/bin`.

2 **Execute a command such as the following:**

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

For example, you can use the following command to add root CA certificate `C:/certificates/CA.crt` into *MIDDLEWARE_HOME*`/jdk160_11/jre/lib/security/cacerts` with `DB_CA_ROOT` as the certificate alias in the keystore. Default storepass (`changeit`) is assumed.

```
keytool -import -alias DB_CA_ROOT -file C:/certificates/CA.crt -keystore ../lib/
security/cacerts -storepass changeit -trustcacerts
```

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

3 **Repeat the procedure for each root CA certificate that you want to import.**

Note: If you obtained all signed certificates from one CA, you need to import the root CA certificate only once.

Configuring and Deploying EPM System

During configuration, you can select the settings that force EPM System components to use SSL communication with the following SSL-enabled components:

- Database
- SMTP mail server
- Application server
- Web server

Selecting SSL settings in EPM System Configurator screens does not SSL-enable your deployment; it only sets a Shared Services Registry flag that forces the use of the secure protocol while communicating with the components for which you specified SSL settings. You must complete manual procedures to ensure that SSL communication is enabled between the components and EPM System.

Note: You must enter or select information on all the configuration screens that EPM System Configurator displays. The following procedure discusses the screens on which SSL settings are specified. See the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

➤ To specify SSL settings for a full SSL deployment of EPM System:

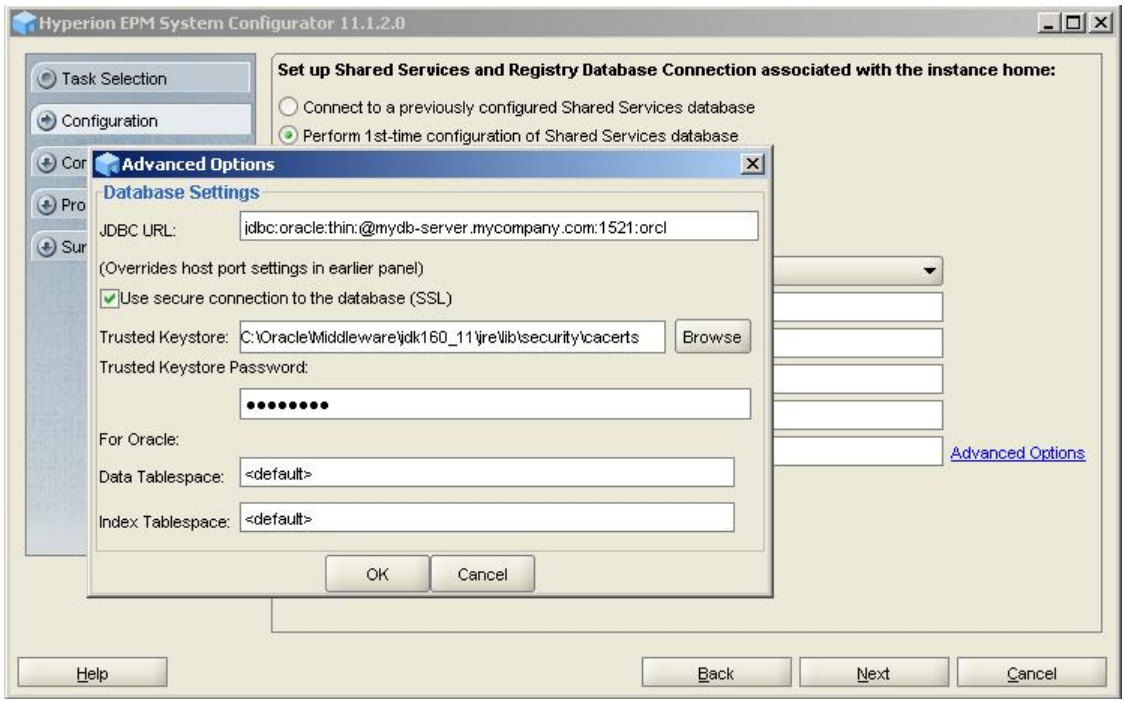
1 Launch EPM System Configurator.

2 Select deployment tasks for all the EPM System components to be deployed on this machine.

3 Click **Next**.

4 Enter database information for Shared Services database and Shared Services Registry:

5 Select the **Advanced Options** link.



6 In the Advanced Options screen, specify the required database SSL configuration parameters through the JDBC URL that EPM System Configurator uses to communicate with the database. EPM System Configurator writes the URL into the Shared Services Registry for use during runtime.

Caution! Before specifying SSL settings for the database, ensure that the database server is configured for SSL.

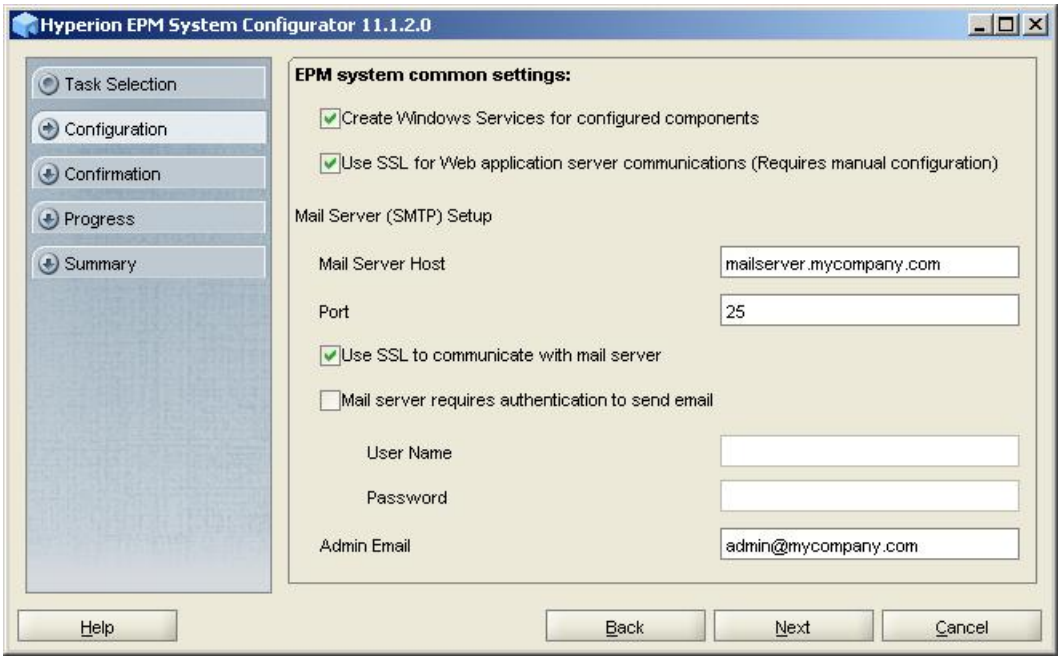
a. Select **Use secure connection to the database**.

b. In **Trusted Keystore**, select the keystore into which you imported the root certificate of the CA that signed the database certificate.

c. If you changed the default keystore password, enter the password in **Trusted Keystore Password**.

d. Click **OK**.

7 Click **Next**.



8 In the EPM System common settings screen, select settings:

Caution! Before selecting the settings to use SSL to communicate with the e-mail server, ensure that the e-mail server is configured for SSL.

a. Select **Use SSL for Web application server communication (Requires manual configuration)** to specify that EPM System should use SSL for communication.

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

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

d. Enter or select settings in the remaining fields.

9 Click **Next**.

The Database configuration screen for the other components you selected for deployment opens. Select an option:

- **Connect to a previously configured database** to use the Shared Services database.
- **Perform 1st-time configuration of the database** to use a new database for the components that you are deploying.

10 **Optional:** If you chose **Perform 1st-time configuration of the database**, enter database connection settings.

11 Select **Advanced Options**, and then select the required options. See step 6.

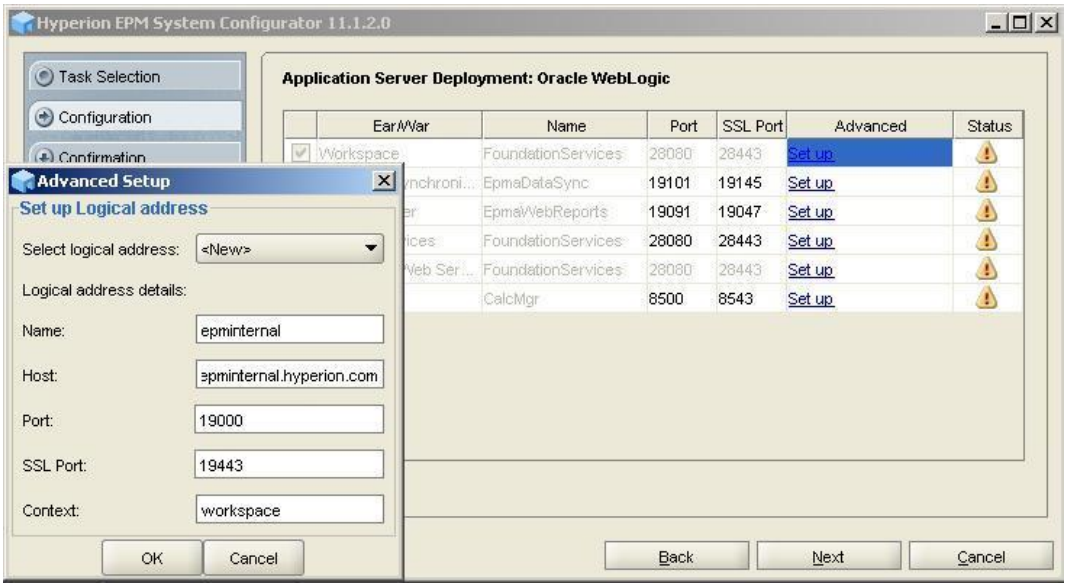
12 Click **Next** until the Application Server Deployment: Oracle WebLogic screen opens.

Note: Ensure that you specify settings on each screen you encounter.

The Application Server Deployment: Oracle WebLogic screen lists the components that you selected to deploy, and the default WebLogic Server deployment settings. To support a full-SSL deployment, you must update the SSL port and host that the components use for server-to-server (internal) communication.

13 **Perform this step for each component listed in Application Server Deployment: Oracle WebLogic screen.**

a. From **Advanced**, select **Set up**.



b. Modify the information in the following fields. See "Advanced Setup" in the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide* for a description of all the fields on this screen.

Table 10 Fields to Modify to Define Logical Addresses

Field	Description
Host	The common name (a unique virtual host name, for example, `epminternal.mycompany.com`) that you used while generating the certificate request for obtaining the certificate for internal communication. This host name must be added as server aliases in the hosts file. See "Adding Server Aliases" on page 37.
Port	Specify the web server port, for example, 19000, that you plan to use for internal communication.
SSL Port	The web server SSL port, for example, 19443, that you plan to use for internal communication.

c. Click **OK**.

14 **Repeat step 13 to create the logical address for each component listed in the screen.**

15 **Click Next until the screen for configuring web server for EPM System components that use IIS (for example, Financial Management) is displayed.**

a. Click **Setup logical address for the web server**.

b. In Set up Logical address screen, modify the information as needed. See Table 10 for the information that you should change.

c. Click **OK**.

16 **Click Next until the deployment process is complete.**

Configuring the Web Server

Subtopics

Configuring SSL Ports and Virtual Hosts

On the Oracle HTTP Server machine, manually configure the SSL ports for internal and external communication. Also, create virtual hosts for internal and external communication.

Caution! *EPM_ORACLE_INSTANCE*/httpConfig/ohs/config/OHS/ohs_component/ssl.conf is updated each time you perform the *Configure Web Server* task in EPM System Configurator. If you reconfigure the web server, verify that the internal and external communication ports are not updated when you configure the web server.

➤ To manually configure web server listen ports:

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

2 **Ensure that the SSL ports are listed under** OHS Listen port**. You should have two entries, similar to the following:**

```
Listen EXTERNAL_SSL_PORT
Listen INTERNAL_SSL_PORT
```

If you are using 4443 as the port for external communication and 19443 as the port for internal communication, your entries should be as follows:

```
Listen 4443
Listen 19443
```

3 **Set** SSLSessionCache **parameter value to** none.

4 **Create two virtual host definitions similar to the following:**

Note: Include the directive proxypreservehost ON in the virtual host definitions if you are front-ending EPM System with an SSL offloader.

```
NameVirtualHost epm.myCompany.com:4443
   <VirtualHost epm.myCompany.com:4443>
     ServerName epm.myCompany.com
        <IfModule ossl_module>
            SSLEngine on
            SSLProxyEngine On
```

```
            SSLVerifyClient None
            SSLCipherSuite
SSL_RSA_WITH_RC4_128_MD5,SSL_RSA_WITH_RC4_128_SHA,SSL_RSA_WITH_3DES_EDE_CBC_SHA,SSL_
RSA_WITH_DES_CBC_SHA,TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256_CBC_SHA
            SSLCRLCheck Off
            SSLWallet "C:\Oracle\middleware\ohs\bin\wallet\epmwallet"
            SSLProxyWallet "C:\Oracle\middleware\ohs\bin\wallet\epmwallet"
            <FilesMatch "\.(cgi|shtml|phtml|php)$">
                SSLOptions +StdEnvVars
            </FilesMatch>
            <Directory "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/$
{COMPONENT_NAME}/cgi-bin">
                SSLOptions +StdEnvVars
            </Directory>
            BrowserMatch ".*MSIE.*" \
            nokeepalive ssl-unclean-shutdown \
            downgrade-1.0 force-response-1.0
        </IfModule>
</VirtualHost>

NameVirtualHost epminternal.myCompany.com:19443
   <VirtualHost epminternal.myCompany.com:19443>
     ServerName epminternal.myCompany.com
        <IfModule ossl_module>
            SSLEngine on
            SSLProxyEngine On
            SSLVerifyClient None
            SSLCipherSuite
SSL_RSA_WITH_RC4_128_MD5,SSL_RSA_WITH_RC4_128_SHA,SSL_RSA_WITH_3DES_EDE_CBC_SHA,SSL_
RSA_WITH_DES_CBC_SHA,TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256_CBC_SHA
            SSLCRLCheck Off
            SSLWallet "C:\Oracle\middleware\ohs\bin\wallet\epminternal"
            SSLProxyWallet "C:\Oracle\middleware\ohs\bin\wallet\epminternal"
            <FilesMatch>>
                SSLOptions +StdEnvVars
            </FilesMatch>
            <Directory "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}/$
{COMPONENT_NAME}/cgi-bin">
                SSLOptions +StdEnvVars
            <Directory >
            BrowserMatch ".*MSIE.*" \
            nokeepalive ssl-unclean-shutdown \
            downgrade-1.0 force-response-1.0
            <Location />
             Options FollowSymLinks
                 AllowOverride None
                 Order Deny,Allow
                 Deny from All
              # Allow from should list all servers that host EPM components
                 Allow from myServer1,myServer2
                 DirectoryIndex welcome-index.html
            </Location>
        </IfModule>
    </VirtualHost>
```

5 **Save and close** `ssl.conf`.

Configuring Redirection to WebLogic Server

Instruct Oracle HTTP Server to redirect internal traffic to the virtual host that is designed to handle internal communication. EPM System Configurator updates this file.

➤ To configure redirection from Oracle HTTP Server to WebLogic Server:

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

2 **Ensure that the** WLSSLWallet **directive points to the Oracle Wallet where the certificates for internal and external communication were imported (see "Configuring the Web Server" on page 44). For example, your directive may be as follows:**

```
WLSSLWallet MIDDLEWARE_HOME/ohs/bin/wallets/myWallet
```

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

3 **Ensure that the value of** SecureProxy **directive is set to** ON.

```
SecureProxy ON
```

4 **Ensure that the** LocationMatch **definitions for deployed EPM System components are similar to the following Shared Services example, which assumes a WebLogic Server cluster (on** myserver1 **and** myserver2**):**

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

5 **Save and close** mod_wl_ohs.conf.

6 **Restart Oracle HTTP Server.**

Configuring EPM System Web Components Deployed on WebLogic Server

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

➤ To configure the web components for SSL:

1 **Start the WebLogic Server by executing a file stored in** *EPM_ORACLE_INSTANCE*/domains/EPMSystem/bin:

- startWebLogic.cmd (Windows)
- startWebLogic.sh (UNIX)

2 **Launch the WebLogic Server Administration Console by accessing the following URL:**

```
http://SERVER_NAME:Port/console
```

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

3 On the Welcome screen, enter the 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 Foundation Services components, you can SSL-enable these servers:

- `CalcManager`
- `EPMADataSync`
- `FoundationServices`
- `EpmaWebReports`

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.

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**:

i. 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.

ii. In **Custom Trust Keystore Type**, enter `jks`.

iii. In **Custom Trust Keystore Passphrase** and **Confirm Custom Trust Keystore Passphrase**, enter the keystore password.

g. Click **Save**.

11 Specify SSL settings.

a. Select **SSL**.

b. In **Private Key Alias**, enter the alias that you specified while importing the signed WebLogic Server certificate.

c. In **Private Key Passphrase** and **Confirm Private Key Passphrase**, enter the password to be used to retrieve the private key.

d. **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.

   i. Select **Advanced**.

   ii. In **Hostname Verification**, select **None**.

e. Click **Save**.

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

Optional: Setting up Redirection from Oracle HTTP Server Load Balancer to Internet Information Services

Generally, in the absence of a standard load balancer, you can configure an Oracle HTTP Server to load balance web applications deployed on multiple IIS Servers. Complete this procedure to redirect requests from the Oracle HTTP Server that is used as the load balancer to the IIS Servers that the following components use:

- Performance Management Architect
- Financial Management
- FDM
- Strategic Finance
- Oracle Hyperion Data Relationship Management, Fusion Edition

The steps in the following procedure assume the following:

- EPM System component, for example, Financial Management, is deployed on two IIS Servers.
- Some EPM System components, for example, Shared Services and EPM Workspace, are deployed on WebLogic Server.
- EPM System components are configured to use Oracle HTTP Server as the web server.

  The logical web application for EPM System components deployed on IIS Server is defined as `http://epm.myCompany.com:19000`.

➤ To configure redirection from Oracle HTTP Server:

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

2 Ensure that the following entry is not commented out. If the entry is missing, add it.

```
LoadModule proxy_balancer_module "${ORACLE_HOME}/ohs/modules/mod_proxy_balancer.so"
```

3 Ensure that the `mod_header` module is loaded by including or uncommenting the following directive.

```
LoadModule headers_module "${ORACLE_HOME}/ohs/modules/mod_headers.so"
```

4 **Create proxy balancer definition for EPM System components (Financial Management, Performance Management Architect, FDM, and Strategic Finance) hosted on IIS Servers by adding definitions similar to the following:**

You must create a `Proxy balancer` definition for each EPM System component deployed on the IIS Server. Within the `Proxy balancer` definition, you must add a `BalanceMember` directive for each IIS Server that hosts EPM System components.

```
<Proxy balancer://iisappshfm>
       BalancerMember http://myIISserver1.mycompany.com:80/hfm
       loadfactor=1 route=1
       BalancerMember http://myIISserver2.mycompany.com:80/hfm
       loadfactor=1 route=2
       ProxySet lbmethod=bytraffic
</Proxy>
<Proxy balancer://iisappshfmlcmserver>
       BalancerMember http://myIISserver1.mycompany.com:80/hfmlcmserver
       loadfactor=1 route=1
       BalancerMember http://myIISserver2.mycompany.com:80/hfmlcmserver
       loadfactor=1 route=2
       ProxySet lbmethod=bytraffic
</Proxy>
<Proxy balancer://iisappshfmsmartviewprovider>
       BalancerMember http://myIISserver1.mycompany.com:80/hfmsmartviewprovider
       loadfactor=1 route=1
       BalancerMember http://myIISserver2.mycompany.com:80/hfmsmartviewprovider
       loadfactor=1 route=2
       ProxySet lbmethod=bytraffic
</Proxy>
<Proxy balancer://iisappshfmapplicationservice>
       BalancerMember http://myIISserver1.mycompany.com:80/hfmapplicationservice
       loadfactor=1 route=1
       BalancerMember http://myIISserver2.mycompany.com:80/hfmapplicationservice
       loadfactor=1 route=2
       ProxySet lbmethod=bytraffic
</Proxy>
<Proxy balancer://iisappshfmlcmservice>
       BalancerMember http://myIISserver1.mycompany.com:80/hfmlcmservice
       loadfactor=1 route=1
       BalancerMember http://myIISserver2.mycompany.com:80/hfmlcmservice
       loadfactor=1 route=2
       ProxySet lbmethod=bytraffic
</Proxy>
<Proxy balancer://iisappsdrm>
    BalancerMember http://myIISserver1.mycompany.com:80/drm-web-client
    loadfactor=1 route=1
    BalancerMember http://myIISserver2.mycompany.com:80/drm-web-client
    loadfactor=1 route=2
ProxySet lbmethod=bytraffic
</Proxy>
```

In the preceding example, values such as *iisappshfm* and `iisappshfmlcmserver` are references and not URLs. `myIISserver1.myCompany.com` and `myIISserver2.myCompany.com` are the names of the IIS Server host machines.

5 **Enable sticky load balancing by adding directives similar to the following. These sample directives instruct Oracle HTTP Server to insert a cookie that keeps track of the route for sticky load balancing of the proxy balancers you defined in step 4.**

```
Header add Set-Cookie "ORA_EPM_IIShfm_ROUTE_ID=iisappshfm .%
{BALANCER_WORKER_ROUTE}e; path=/hfm ;" env=BALANCER_ROUTE_CHANGED
Header add Set-Cookie
"ORA_EPM_IIShfmofficeprovider_ROUTE_ID=iisappshfmofficeprovider.%
{BALANCER_WORKER_ROUTE}e; path=/hfmofficeprovider;" env=BALANCER_ROUTE_CHANGED
Header add Set-Cookie "ORA_EPM_IIShfmlcmserver_ROUTE_ID=iisappshfmlcmserver.%
{BALANCER_WORKER_ROUTE}e; path=/hfmlcmserver;" env=BALANCER_ROUTE_CHANGED
Header add Set-Cookie
"ORA_EPM_IIShfmsmartviewprovider_ROUTE_ID=iisappshfmsmartviewprovider.%
{BALANCER_WORKER_ROUTE}e; path=/hfmsmartviewprovider;" env=BALANCER_ROUTE_CHANGED
Header add Set-Cookie
"ORA_EPM_IIShfmapplicationservice_ROUTE_ID=iisappshfmapplicationservice.%
{BALANCER_WORKER_ROUTE}e; path=/hfmapplicationservice;" env=BALANCER_ROUTE_CHANGED
Header add Set-Cookie "ORA_EPM_IIShfmlcmservice_ROUTE_ID=iisappshfmlcmservice.%
{BALANCER_WORKER_ROUTE}e; path=/hfmlcmservice;" env=BALANCER_ROUTE_CHANGED
Header add Set-Cookie "ORA_EPM_IIS_ROUTE_ID=iisapps.%{BALANCER_WORKER_ROUTE}e;
path=/;" env=BALANCER_ROUTE_CHANGED
Header add Set-Cookie "BALANCEID= iisappsdrm.%
{BALANCER_WORKER_ROUTE}e; path=/drm-web-client;" env=BALANCER_ROUTE_CHANGED
```

6 **Replace the existing `ProxyPass` directive for EPM System components (Financial Management, Performance Management Architect, FDM, Oracle Hyperion Data Relationship Management, Fusion Edition, and Strategic Finance) deployed on IIS Server with definitions similar to the following:**

```
ProxyPass /hfm balancer://iisappshfm/ stickysession=ORA_EPM_IIShfm_ROUTE_ID
nofailover=On
ProxyPassReverse /hfm http://epm.myCompany.com:19000/hfm
ProxyPreserveHost ON

ProxyPass /hfmofficeprovider balancer://iisappshfmofficeprovider/
stickysession=ORA_EPM_IIShfmofficeprovider_ROUTE_ID nofailover=On
ProxyPassReverse /hfmofficeprovider http://epm.myCompany.com:19000/hfmofficeprovider
ProxyPreserveHost ON

ProxyPass /hfmlcmserver balancer://iisappshfmlcmserver/
stickysession=ORA_EPM_IIShfmlcmserver_ROUTE_ID nofailover=On
ProxyPassReverse /hfmlcmserver http://epm.myCompany.com:19000/hfmlcmserver
ProxyPreserveHost ON

ProxyPass /hfmsmartviewprovider balancer://iisappshfmsmartviewprovider/
stickysession=ORA_EPM_IIShfmsmartviewprovider_ROUTE_ID nofailover=On
ProxyPassReverse /hfmsmartviewprovider http://epm.myCompany.com:19000/
hfmsmartviewprovider
ProxyPreserveHost ON

ProxyPass /hfmapplicationservice balancer://iisappshfmapplicationservice/
stickysession=ORA_EPM_IIShfmapplicationservice_ROUTE_ID nofailover=On
ProxyPassReverse /hfmapplicationservice http://epm.mycompany.com:19000/
hfmapplicationservice
ProxyPreserveHost ON

ProxyPass /hfmlcmservice balancer://iisappshfmlcmservice/
stickysession=ORA_EPM_IIShfmlcmservice_ROUTE_ID nofailover=On
ProxyPassReverse /hfmlcmservice http://epm.mycompany.com:19000/hfmlcmservice
```

```
ProxyPreserveHost ON

ProxyPass /drm-web-client balancer://iisappsdrm stickysession=BALANCEID
nofailover=Off
ProxyPassReverse /drm-web-client http://epm.mycompany.com:19000/drm-web-client
ProxyPassReverse /drm-web-client http://epm.mycompany1.com:19000/drm-web-client
```

7 **Save and close** `httpd.conf`.

Testing the Deployment

After completing the SSL deployment, verify that everything works.

➤ To test your deployment:

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

`https://`*`WEB_SERVER_NAME`*`:`*`SSL_PORT`*`/workspace/index.jsp`

In the URL, replace *`WEB_SERVER_NAME`* with the server alias that you created for external communication (see "Adding Server Aliases" on page 37) and *`SSL_PORT`* with the SSL web server port for external communication. For example, if you used `epm.myCompany.com` as the server alias for external communication and `4443` as the SSL port, your URL is:

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

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

3 **Click Log On.**

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

Optional: Reconfiguring Oracle HTTP Server

If you configured and deployed EPM System components in multiple deployment sessions, you must perform the web server deployment task to add the component contexts from the Shared Services Registry to the Oracle HTTP Server configuration files.

➤ To update web server configuration:

1 **Launch EPM System Configurator.**

2 **On task selection screen, select Configure Web Server from the Foundation Services deployment options.**

3 **Click Next.**

4 **Update the web server configuration. See "Configuring the Web Server" on page 44.**

Configuring SSL-enabled External User Directories

Subtopics

Assumptions

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

Import root CA certificate

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

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

- WebLogic Server:
  - **Sun JVM keystore:** *MIDDLEWARE_HOME*/jdk160_11/jre/lib/security/cacerts
  - **JRockit JVM keystore:** *MIDDLEWARE_HOME*/jrockit_160_05/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/jdk160_11/jre/lib/security/cacerts

Configure External User Directories

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

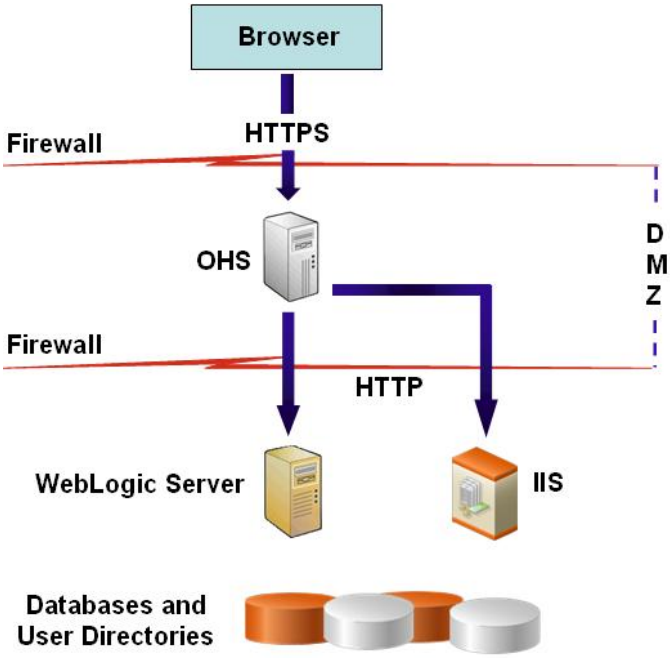
See "Configuring User Directories" in the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Terminating SSL at the Web Server

Subtopics

Deployment Architecture

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



Note: Oracle HTTP Server uses `mod_wl_ohs` for redirection to WebLogic Server and mod_proxy for redirection to IIS.

Assumptions

This discussion assumes that EPM System components are installed on the server but are not configured. In addition to the EPM System components that you select, the default EPM System installation process installs WebLogic Server and Oracle HTTP Server within *MIDDLEWARE_HOME*.

Process Overview

The following steps are involved in configuring SSL:

1. Install the EPM System. See "Installing EPM System" on page 32.
2. Add server aliases on the web server and SSL-enable Oracle HTTP Server. See "Adding Server Aliases" on page 54.
3. Install the Oracle HTTP Server certificate. See "Installing Certificates for Oracle HTTP Server" on page 54.
4. Using EPM System Configurator, configure and deploy EPM System components. See "Configuring and Deploying EPM System" on page 55.
5. Configure Oracle HTTP Server. See "Configuring Ports and Virtual Hosts on Oracle HTTP Server" on page 56.
6. Test your deployment. See "Testing the Deployment" on page 58.

Configuring EPM System

Subtopics

Adding Server Aliases

This scenario uses two server aliases, one for SSL-enabled external communication between Oracle HTTP Server and browsers (for example, `epm.myCompany.com`), and the other for routing internal non-SSL communication among EPM System servers (for example, `empinternal.myCompany.com`). Oracle recommends that you use a certificate tied to the server alias for external communication to prevent exposing server names to users.

You add server aliases in the `hosts` file on the machine. Ensure that the server aliases point to the IP address of the machine that hosts Oracle HTTP Server. Clients must resolve these aliases through DNS.

Installing Certificates for Oracle HTTP Server

Oracle HTTP Server is installed as the default web server when you install EPM System components. SSL certificates for Oracle HTTP Server are stored in Oracle Wallet.

You need one signed certificate for external communication to support SSL termination at Oracle HTTP Server. Oracle recommends that this certificate be tied to a server alias, for example, `epm.myCompany.com`, to enhance security. If you are not using a third-party CA known to Oracle HTTP Server, you also need the root CA certificate.

Note: Perform this procedure on each Oracle HTTP Server host machine.

See "Creating Wallets and Installing Certificates for Oracle HTTP Server" on page 38.

Configuring and Deploying EPM System

During configuration, you can select the settings that force EPM System components to use SSL communication with Oracle HTTP Server.

Selecting SSL settings in EPM System Configurator screens does not SSL-enable your deployment. You must complete manual procedures to enable SSL communication.

Note: You must enter or select information in all the configuration screens that EPM System Configurator displays. The following procedure discusses the screens where SSL settings are specified. For detailed information on specifying information in EPM System Configurator screens, see the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

➤ To specify settings for SSL termination at the web server:

1 **Launch EPM System Configurator.**

2 **Select deployment tasks for the EPM System components that you plan to deploy on this machine.**

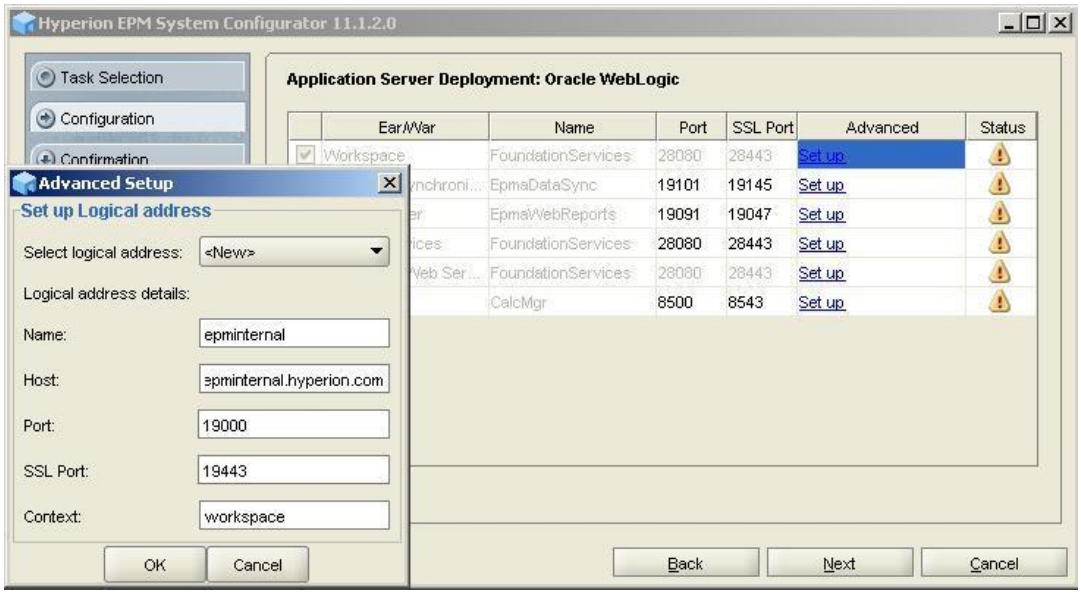
3 **Click Next until the Application Server Deployment: Oracle WebLogic screen opens.**

Note: Ensure that you specify settings on each screen that you encounter.

Application Server Deployment: Oracle WebLogic screen lists the EPM System components that you selected to deploy, and the default WebLogic Server deployment settings. To support SSL termination at the web server, you must force the components to use a non-SSL port for internal communication.

4 **Perform this step for each component listed on the Application Server Deployment: Oracle WebLogic screen.**

a. From **Advanced**, select **Set up**.



b. Modify the information in the following fields. See "Advanced Setup" in the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide* for a description of all the fields on this screen.

Table 11 Fields to Modify to Define Logical Addresses

Field	Description
Host	The common name (a server alias, for example, `epminternal.mycompany.com`) that you used while generating the certificate request for obtaining the certificate for internal communication. This host name must be added as server aliases in the hosts file. See "Adding Server Aliases" on page 54.
Port	Specify the non-SSL web server port, for example, 19000, that you plan to use for internal communication.

c. Click **OK**.

5 **Repeat step 4 to create the logical address for each component listed in the screen.**

6 **Click Next until the screen for configuring web server for EPM System components that use IIS (for example, Financial Management) is displayed.**

a. Click **Setup logical address for the web server**.

b. On the Set up Logical address screen, modify the information as needed. See Table 11 for the information that you should change.

c. Click **OK**.

7 **Click Next until the deployment process is complete.**

Configuring Ports and Virtual Hosts on Oracle HTTP Server

On the Oracle HTTP Server machine, manually configure the ports for internal and external communication. Also, create virtual hosts for internal and external communication.

Configuring SSL Port and Virtual Host for External Communication

➤ To configure the port and virtual host for external communication:

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

2 **Ensure that the SSL port for external communication is listed under** `OHS Listen port`**. The entry should be similar to the following:**

```
Listen EXTERNAL_SSL_PORT
```

If you are using `4443` as the port for external communication, your entry should be:

```
Listen 4443
```

3 **Create a virtual host definition similar to the following:**

Note: Include the directive `proxypreservehost ON` if you are front-ending EPM System with an SSL offloader.

```
NameVirtualHost epm.myCompany.com:4443
   <VirtualHost epm.myCompany.com:4443>
     ServerName epm.myCompany.com
        <IfModule ossl_module>
            SSLEngine on
            SSLProxyEngine On
            SSLVerifyClient None
            SSLCipherSuite SSL_RSA_WITH_RC4_128_MD5,SSL_RSA_WITH_RC4_128_SHA,
              SSL_RSA_WITH_3DES_EDE_CBC_SHA,SSL_RSA_WITH_DES_CBC_SHA,
               TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256_CBC_SHA
            SSLCRLCheck Off
            SSLWallet "C:\Oracle\middleware\ohs\bin\wallet\epmwallet"
            SSLProxyWallet "C:\Oracle\middleware\ohs\bin\wallet\epmwallet"
            <FilesMatch "\.(cgi|shtml|phtml|php)$">
               SSLOptions +StdEnvVars
            </FilesMatch>
            <Directory "${ORACLE_INSTANCE}/config/${COMPONENT_TYPE}
               /${COMPONENT_NAME}/cgi-bin">
               SSLOptions +StdEnvVars
            </Directory>
            BrowserMatch ".*MSIE.*" \
            nokeepalive ssl-unclean-shutdown \
            downgrade-1.0 force-response-1.0
        </IfModule>
</VirtualHost>
```

4 **Save and close** `ssl.conf`.

Configuring Port and Virtual Host for Internal Communication

➤ To configure the port and virtual host for internal communication:

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

2 **Ensure that the port for internal communication is listed as the value of** `OHS Listen port`**. The entry should be similar to the following:**

```
Listen INTERNAL_COMM_PORT
```

If you are using `19000` as the port for internal communication, your entry should be:

```
Listen 19000
```

> **Caution!** EPM System Configurator resets the listen port number to the default value (`19000`) each time you reconfigure the web server. If you are not using the default port, verify the port number in `httpd.conf`.

3 **Create a virtual host definition similar to the following:**

> **Note:** Include the directive `proxypreservehost ON` if you are front-ending EPM System with an SSL offloader.

```
NameVirtualHost epminternal.myCompany.com:19000
   <VirtualHost epminternal.myCompany.com:19000>
     ServerName epminternal.myCompany.com
   </VirtualHost>
```

4 **Save and close** `httpd.conf`.

Configuring Oracle HTTP Server Communication with WebLogic Server

➤ To configure communication with WebLogic Server:

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

2 **Ensure that the** `WLProxySSL` **directive is enabled, for example:**

```
WLProxySSL ON
```

3 **Save and close** `mod_wl_ohs.conf`.

4 **Restart Oracle HTTP Server.**

Testing the Deployment

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

`https://`*WEB_SERVER_NAME*`:`*SSL_PORT*`/workspace/index.jsp`

See "Testing the Deployment" on page 51.

Deploying EPM System with an SSL Offloader

Subtopics

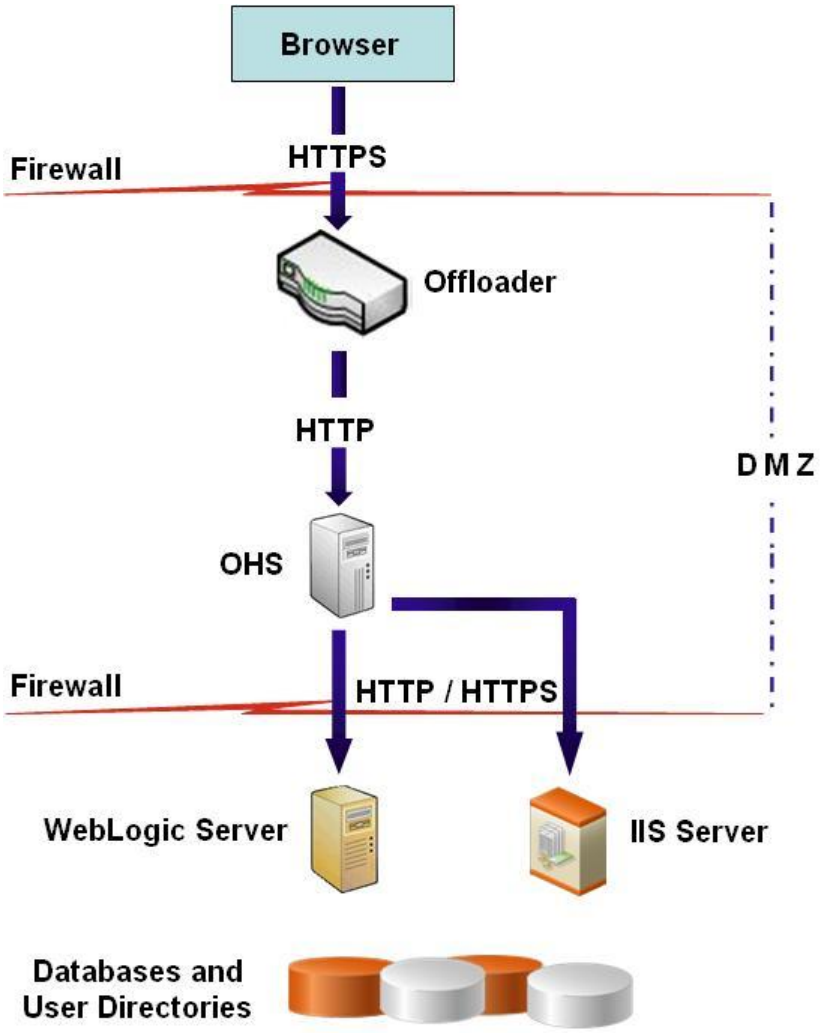
Deployment Architecture

In this enterprise topology, the SSL connection can be terminated at the offloader (similar to terminating SSL at the web server) or can extend beyond the offloader (similar to full SSL) The offloader accepts encrypted requests from the browser and decrypts them. If SSL is terminated at the offloader, unencrypted data is passed from the offloader to Oracle HTTP Server, which is configured with WebLogic Server plugin. An optional load balancer can be used to route traffic between the offloader and multiple Oracle HTTP Servers. Oracle HTTP Server routes requests to EPM System components deployed on WebLogic Server or IIS Server. Server-to-server communication is routed through the web server without offloader involvement.

Based on security requirements, you can use SSL for communication between Oracle HTTP Server and the deployed EPM System components, including databases and user directories.

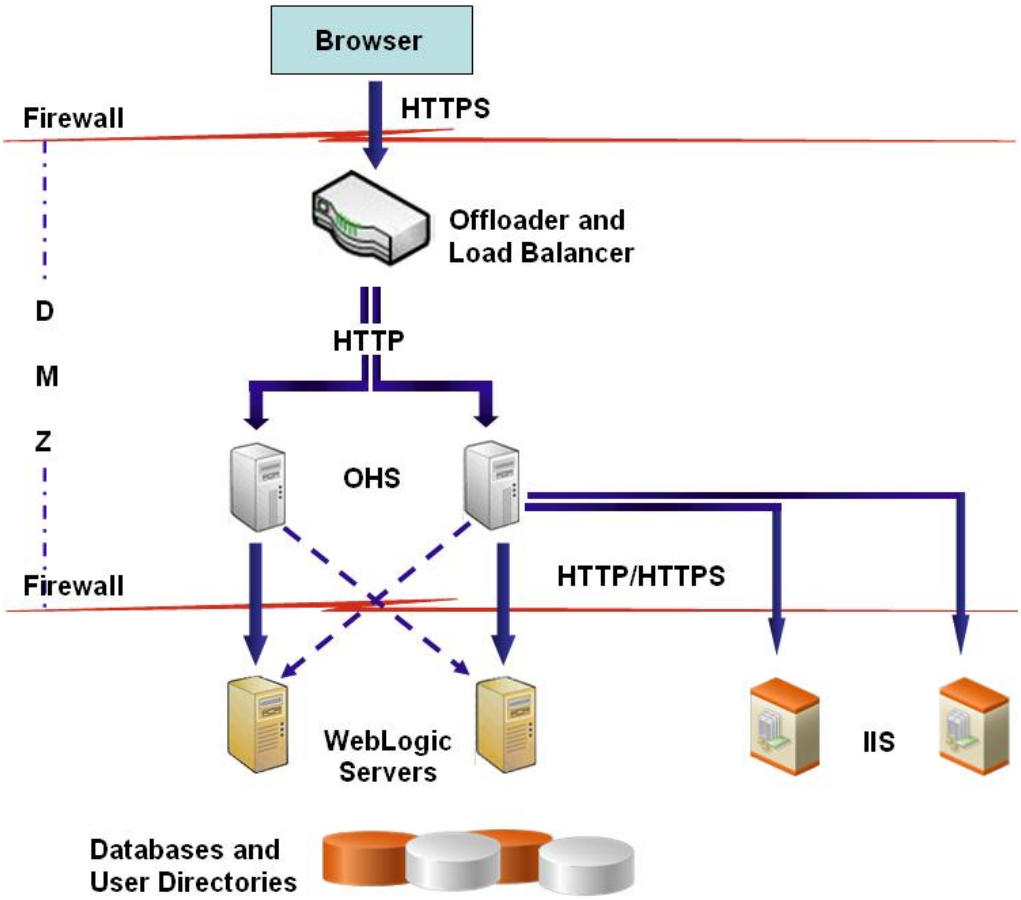
If you choose not to use SSL for communication between Oracle HTTP Server and the deployed EPM System components, you can minimize security risks by deploying the offloader and Oracle HTTP Server in the DMZ behind a firewall on a secure subnet to which users do not have direct access. WebLogic Servers, IIS Servers, and other components could be behind another firewall to ensure greater security.

The following illustration presents a conceptual deployment using one Oracle HTTP Server:



Note: Oracle HTTP Server uses `mod_wl_ohs` for redirection to WebLogic Server and mod_proxy for redirection to IIS.

The following illustration presents a conceptual high availability deployment using a load balancer and SSL accelerator:



Required Load Balancer Features

The load balancer that you select must have the following features:

- Load-balancing of traffic to a pool of real servers through a virtual host name. Browsers access services using the virtual host name, and the load balancer forwards the requests to the servers in the pool.
- Port translation configuration support where requests are forwarded to a back-end server port that is different from the port in the request.
- Port monitoring on the servers in the pool to determine availability of a service.
- Virtual servers and port configuration, the ability to configure server aliases and ports that meet the following requirements on the external load balancer:
  - Support for multiple virtual server configurations

    For each virtual server, the load balancer should allow configuration of traffic management on more than one port. For example, for Oracle HTTP Server, the load balancer should support a virtual server and port each for HTTP and HTTPS traffic.
  - Ability to associate each server alias with an IP address and be a part of the DNS. Clients must be able to access the external load balancer through server alias.
- Node failure detection and the ability to stop routing traffic to a failed node

- Fault-tolerant mode. Oracle recommends that the load balancer be configured in fault-tolerant mode.
- Sticky routing, for example, cookie-based and IP-based persistence.
- SSL acceleration. The load balancer should terminate SSL traffic and forward unencrypted traffic to the back-end servers using the equivalent non-SSL protocol.

Oracle recommends that you configure the load balancer virtual server to return immediately to the calling client when the back-end services to which it forwards traffic are unavailable. This configuration is preferred over the client disconnecting on its own after a timeout based on the TCP/IP settings on the client machine.

Assumptions

- EPM System components are installed on the server but are not configured. In addition to the EPM System components that you select, the default EPM System installation process installs WebLogic Server and Oracle HTTP Server within *MIDDLEWARE_HOME*.
- The offloader, and load balancer, if you are using one, are fully integrated into your deployment environment.
- Communication between the browser and offloader is SSL-enabled.

Process Overview

The following steps are involved in configuring SSL with an offloader:

1. Configure the offloader using the documentation from the offloader vendor. Configure the offloader to receive SSL communication from clients through a secure port and decrypt it.
   - Add server aliases on the offloader. See "Adding a Server Alias" on page 63.
   - If you are not using a load balancer, the offloader should forward the decrypted communication to Oracle HTTP Server.
   - If you are using a load balancer, the offloader should forward the decrypted communication to the load balancer, which routes communication to Oracle HTTP Servers.
2. Install EPM System. See "Installing EPM System" on page 32.
3. Depending on your security requirements, configure and deploy EPM System components using EPM System Configurator.
   - If you are terminating SSL at the offloader, see "SSL Terminated at Offloader" on page 63.
   - If you are not terminating SSL at the offloader, see "Full SSL Deployment of EPM System" on page 32 for detailed deployment procedures for all EPM System components that you can SSL-enable.

Configuring EPM System for SSL Offloading

Subtopics

Adding a Server Alias

Configuring EPM System with an SSL offloader may use a server alias, for example, `epm.myCompany.com`, created on the offloader for external communication. The use of a server alias prevents exposing the physical offloader name to users. The server alias you add must be DNS resolvable and must point to the IP address of the offloader. Users access a secure URL tied to the server alias and SSL-enabled port, for example, `https://epm.myCompany.com:4443/workspace/index.jsp` to access EPM System components.

Configuring and Deploying EPM System

Subtopics

SSL Terminated at Offloader

If you plan to terminate SSL communication at the offloader, configure and deploy EPM System components without selecting SSL settings. You must configure SSL communication between the browser and the offloader (to the virtual offloader alias) separately using the documentation of your offloader vendor.

See the following information sources to deploy EPM System components:

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

Note: You must enter or select information in all the configuration screens that EPM System Configurator displays. The following procedure discusses the screens where you must specify settings to support this topology.

➤ To deploy EPM System components:

1 **Launch EPM System Configurator.**

2 Select deployment tasks for all the EPM System components that you plan to deploy on this machine.

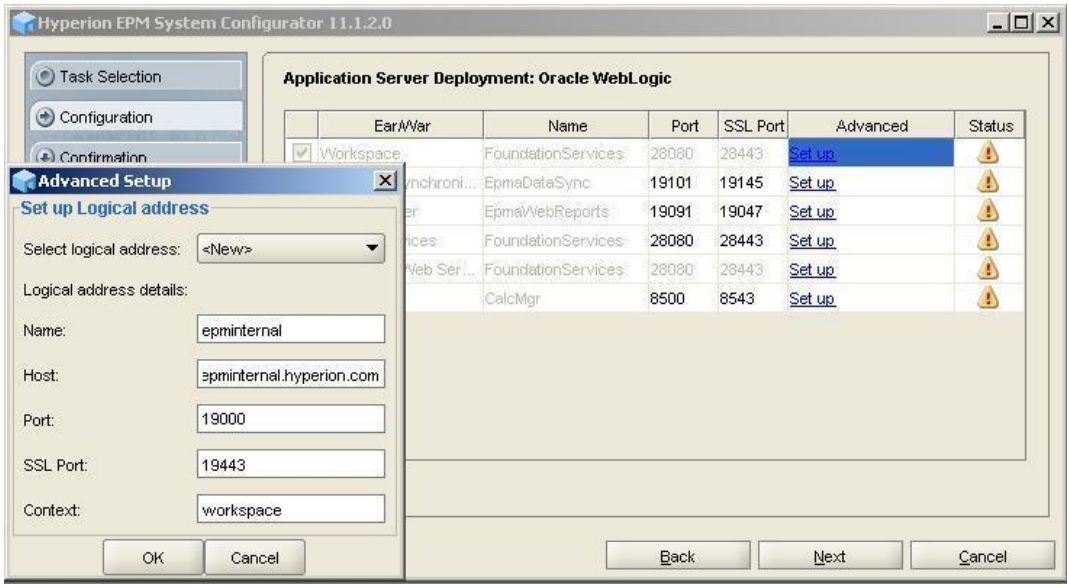
3 Click **Next**.

4 Click **Next** until the Application Server Deployment: Oracle WebLogic screen opens.

Application Server Deployment: Oracle WebLogic screen lists the components that you selected to deploy and the default WebLogic Server deployment settings. To support an SSL offloader deployment, you must update the port and host that the components use for server-to-server (internal) communication.

5 Perform this step for each component listed in Application Server Deployment: Oracle WebLogic screen.

a. From **Advanced**, select **Set up**.



b. Modify the information in the following fields. See "Advanced Setup" in the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide* for a description of the fields in this screen.

Table 12 Fields to Modify to Define Logical Addresses

Field	Description
Host	The name of Oracle HTTP Server host for server-to-server communication, for example, `myWebServer.mycompany.com`.
Port	The web server port, for example, 19000, that you plan to use for server-to-server communication.

c. Click **OK**.

6 Repeat step 5 to create the logical address for each component listed in the screen.

7 Click **Next** until the deployment process is complete.

Optional: Configuring Oracle HTTP Server Communication with WebLogic Server

Complete this procedure if you are terminating SSL at the offloader.

➤ To configure Oracle HTTP Server communication with WebLogic Server:

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

2 **Ensure that the** `WLProxySSL` **directive is enabled, for example:**

```
WLProxySSL ON
```

3 **Save and close** `mod_wl_ohs.conf`.

4 **Restart Oracle HTTP Server.**

Full SSL with Offloader

In this scenario, you can SSL-enable any communication channel beyond the offloader. For example, you can SSL-enable downstream communication between all EPM System components such as Oracle HTTP Server, WebLogic Server, IIS server, and database server. See "Full SSL Deployment of EPM System" on page 32 for a detailed discussion of the steps involved.

Configuring the Load Balancer and Offloader

Complete the following tasks on the offloader hardware using the documentation provided by your offloader hardware vendor:

- SSL-enable a port on the offloader. This port, with the server alias that you created (see "Adding a Server Alias" on page 63), can be used for external (browser to offloader) communication.

  The offloader should unencrypt the requests received by the virtual server and forward the unencrypted requests to Oracle HTTP Server.

- Configure redirection of requests from the offloader:
  - If you are not using a load balancer, configure redirection from the offloader to Oracle HTTP Server.
  - If you are using a load balancer, configure redirection from the offloader to the load balancer.
- If you are using a load balancer, configure it.

Note: In an SSL terminated at load balancer scenario, the offloader must set the value of `WLProxySSL` header to `true` to ensure communication between the SSL-enabled offloader and EPM System that is not SSL-enabled.

If SSL context is lost before the request reaches the Weblogic plugin, you must add a header at the SSL termination point. If you are using BIG-IP, do so by adding `WL-Proxy-SSL:true` as the value in Request Header Insert row while creating a new HTTP profile. See http://www.f5.com/pdf/deployment-guides/f5-weblogic10-dg.pdf for details.

Reporting and Analysis Procedures

Because Oracle's Hyperion Reporting and Analysis Framework web application is not aware of the SSL offloader, You must manually set the Custom SmartCut URL setting to point to the offloader.

➤ To update Custom SmartCut URL:

1. Log into EPM Workspace as Administrator.
2. Select **Navigate**, then **Administer**, then **Reporting and Analysis**, and then **Web Applications**.
3. Right-click **Reporting and Analysis Web Application**, and then **Properties**.
4. Select **Internal**.
5. In General section, edit the value of **Custom SmartCut URL** so that it points to the URL that the offloader uses for external communication; for example, `https://epm.myCompany.com:4443`.

   Note: If you changed EPM Workspace deployment context on the application server, for example, from `/workspace` to `/myCompany/workspace`, be sure to use a URL that reflects the updated context, for example; `https://epm.myCompany.com:4443/myCompany/workspace`.

6. Save your changes.

Testing the Deployment

After completing the SSL deployment, verify that everything works.

➤ To test your deployment:

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

   `https://`*`OFFLOADER_NAME`*`:`*`SSL_PORT`*`/workspace/index.jsp`

   In the URL, replace *`OFFLOADER_NAME`* with the server alias that you created on the offloader for external communication (see "Adding a Server Alias" on page 63) and *`SSL_PORT`* with the SSL port for external communication. For example, if you used `epm.myCompany.com` as the server alias and `4443` as the SSL port, your URL is:

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

2. On the Logon screen, enter a user name and password.
3. Click **Log On**.
4. Verify that you can securely access the deployed EPM System components.

Client Certificate Authentication (Two-Way SSL)

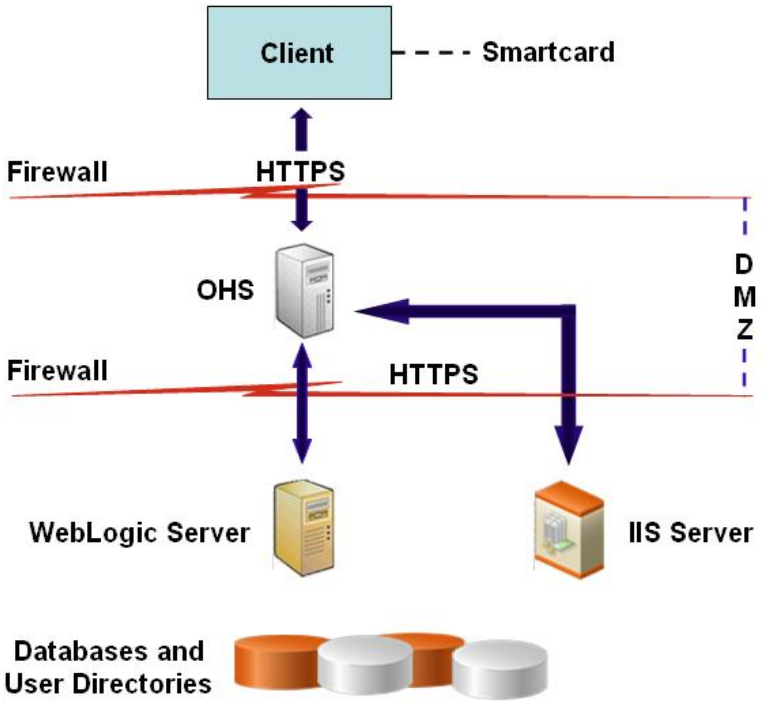
Subtopics

Client Certificate Authentication is supported between clients (browser or Smart View client) and the Oracle HTTP Server web server.

Note: Administration Services Console does not support client certificate authentication.

Deployment Architecture

Client certificate authentication implements two levels of security; Oracle HTTP Server establishes two-way trust between client and server, and Shared Services security verifies user identities based on client certificate. The illustrated concept:



Note: Oracle HTTP Server uses `mod_wl_ohs` for redirection to WebLogic Server and mod_proxy for redirection to IIS.

The simplest implementation of client certificate authentication is when it is used for external communication (between the client and the web server or offloader) only. You can, however, enable two-way SSL among underlying EPM System components as well.

In a client certificate authentication implementation, a personal certificate (x509) issued to an EPM System user is used to authenticate the user. The personal certificate can be installed on a client machine or a smart card. Users are not prompted for a password.

Trust between components is established using the root certificate of the CA that the signed the component's certificate. You must ensure that the root CA certificate of a component is installed on each components with which it communicates. For example, to ensure SSL communication between Oracle HTTP Server and Foundation Services deployed on WebLogic Server, the root certificate of the CA that signed the WebLogic Server certificate must be installed on Oracle HTTP Server, and the root certificate of the CA that signed the Oracle HTTP Server certificate must be installed on WebLogic Server.

Note: If you are not using a trusted third-party CA, to simplify the deployment process, Oracle recommends that you use the same CA to sign all server certificates. Self-signed certificates cannot be used to support client certificate authentication.

Implementing client certificate authentication involves completing manual steps. See "Configuring Client Certificate Authentication in EPM System" on page 71.

Assumptions

- EPM System components are installed on the server but are not configured.
- Your deployment architecture comprises the following servers:
  - A web server running Oracle HTTP Server, which proxies HTTP requests to the application servers.
  - One or more computers that host WebLogic Server, which runs EPM System components such as Planning and Foundation Services.
  - One or more computers that host Internet Information Services (IIS), which runs the following components:
    - Performance Management Architect
    - Financial Management
    - FDM
    - Strategic Finance
  - A server that hosts a database; for example, Oracle Database.
  - A server that hosts a supported user directory, such as Sun Java System Directory Server or Microsoft Active Directory.
- You are using signed certificates from a trusted third-party CA.

If you are using your own CA to sign certificates, Oracle assumes that you are well-versed in generating certificate requests and signing your own certificates.

Caution! Do not use self-signed certificates to set up two-way SSL.

Process Overview

The following steps are involved in configuring two-way SSL:

1. Install EPM System. See "Installing EPM System" on page 32.
2. Depending on your security requirements, complete an SSL deployment or deploy EPM System without SSL communication among servers:
   - "Full SSL Deployment of EPM System" on page 32
   - "Deploying EPM System with an SSL Offloader" on page 59
   - "SSL Terminated at Offloader" on page 63
3. If you are not using certificates from a well-known third-party CA, import root CA certificate as needed. Generally, you should import root CA certificate of a component into the keystore used by each component with which it communicates.
4. Complete custom Oracle HTTP Server configuration. See "Configuring Oracle HTTP Server" on page 69.
5. Complete custom WebLogic Server configuration. See "Configuring Web Components" on page 70.
6. Install certificates for clients (browsers). "Optional: Installing Client Certificates on a Machine" on page 70
7. **Optional:** implement automated authentication. See "Configuring Client Certificate Authentication in EPM System" on page 71.

Configuring Oracle HTTP Server

Update the `ssl.conf` file of Oracle HTTP Server to enable client certificate authentication . You should complete this procedure in addition to configuring Oracle HTTP Server as discussed in the SSL procedures appropriate for your implementation. See:

- "Full SSL Deployment of EPM System" on page 32
- "Deploying EPM System with an SSL Offloader" on page 59
- "SSL Terminated at Offloader" on page 63

➤ To configure client certificate authentication on Oracle HTTP Server:

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

2 **Ensure that the** `SSLVerifyClient` **parameter and** `SSLOptions` **directive are set:**

```
SSLVerifyClient require
SSLOptions +StdEnvVars +ExportCertData
```

3 Include the following directive:

```
RequestHeader set HYPLOGIN "%{SSL_CLIENT_CERT}e"
```

4 Save and close `ssl.conf`.

Configuring Web Components

Perform this procedure for each application deployed on WebLogic Server to configure two-way SSL. Complete this procedure after completing the SSL procedures listed in one of these sections. You do not need to perform this procedure if you are terminating SSL at the web server or offloader.

- "Full SSL Deployment of EPM System" on page 32
- "SSL Terminated at Offloader" on page 63

Before configuring two-way SSL, ensure that the trust keystore contains the root certificate of the CA that signed the Oracle HTTP Server certificate.

➤ To configure two-way SSL for a web component:

1 Log on to WebLogic Server Administration Console.

2 In the Change Center, click **Lock & Edit**.

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

4 In summary of Servers, select the server that you want to SSL-enable.

5 Select **SSL** to open the SSL tab.

6 Click **Advanced** at the bottom of the SSL tab.

7 Select **Client Certs Requested and Enforced** as the two-way SSL behavior. This selection forces the client to present a certificate. If a certificate is not presented, the SSL connection is terminated.

8 Click **Save**.

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

Optional: Installing Client Certificates on a Machine

Complete this procedure if you are installing a personal certificate on a client computer. If you are using a smart card, see the documentation from the smart card vendor for information on installing a certificate on the smart card.

A client certificate (issued to an EPM System user) is required for each user to access EPM System components. The personal certificate supports client certificate authentication between the client and Oracle HTTP Server.

You use your browser's certificate management functionality to import and manage certificates.

Firefox 3.x

➤ To import personal certificates: Firefox 3.x:

1 Select **Tools**, then **Options**, then **Advanced**, and then **Encryption**.

2 Click **View Certificates**.

3 In Certificate Manager, select **Your Certificates**, and then **Import**.

4 In **File Name to Restore**, select a personal certificate (PKCS12) file.

5 Click **Open**.

6 Click **OK** to close the Certificate Manager.

Internet Explorer 7.x

➤ To import personal certificates: Internet Explorer 7.x:

1 Select **Tools**, then **Internet Options**, then **Content**, and then **Certificates**.

2 In Certificates, click **Import**.

3 Follow the instructions on the Certificate Import Wizard to import a personal certificate (PKCS12).

Configuring Client Certificate Authentication in EPM System

Subtopics

Perform this procedure to enable automated user login using a smart card. EPM System requires that an HTTP header named `HYPLOGIN` be used to pass the x509 certificate in PEM format to EPM System security (through the web server or offloader to the application server).

When users access a secure EPM System URL, a two-way secure communication channel is created between the client and the web server. After secure communication is established, the x509 personal certificate is carried in the request header from the web server to the application server to authenticate the user for SSO. The default EPM System authentication mechanism reads the user identity from the DN attribute of the certificate.

Configuring Client Certificate Authentication involves these tasks:

- "Configuring User Directory" on page 72
- "Configuring EPM System SSO" on page 72

Configuring User Directory

EPM System security should be able to verify the user credentials against a configured user directory. The user name in the user directory must match the identity provided through the personal 509 certificate that the client presents to the web server.

You configure user directories using Shared Services Console. See the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Configuring EPM System SSO

You must set security options in Shared Services to force EPM System to use the custom login class to authenticate users. The following procedure explains only the steps that you must take to enable the use of the custom login class. See "Setting Security Options" in the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

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

If you must derive user identity from a certificate attribute other than DN, you must develop and implement a custom login class. See Appendix B, "Implementing a Custom Login Class."

➤ To enable the use of custom login class:

1. **Launch Shared Services Console.**
2. **Select Administration, and then Configure User Directories.**
3. **Select Security Options.**
4. **In Security Options, set global parameters.**
   a. Select **Enable SSO**.
   b. In **SSO Provider or Agent**, select **Other**
   c. In **SSO Mechanism**, select **Custom Login Class**.
   d. In the field next to **SSO Mechanism**, enter `com.hyperion.css.sso.agent.X509CertificateSecurityAgentImpl`.

      If you created a custom login class, you must enter the fully qualified name of the custom class.
5. **Click Save.**
6. **Restart Shared Services and other EPM System products.**

Testing the Deployment

➤ To test Two-way SSL deployment:

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

   `https://`*`WEB_SERVER_NAME`*`:`*`SSL_PORT`*`/workspace/index.jsp`

In the URL, replace *WEB_SERVER_NAME* with the server alias that you created for external communication (see "Adding Server Aliases" on page 37) and *SSL_PORT* with the SSL web server port for external communication. For example, if you used `epm.myCompany.com` as the server alias for external communication and `4443` as the SSL port, your URL is:

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

Note: If you configured transparent client certificate authentication for automated login, EPM Workspace opens. If you are not configured for automated login, the Logon screen opens.

2. **On the Logon screen, enter a user name and password.**
3. **Click Log On.**
4. **Verify that you can securely access the deployed EPM System components.**

Enabling Encryption for Financial Reporting RMI Service

Financial Reporting uses Remote Method Invocation (RMI) to support communication among components such as Financial Reporting application server, Print Services, and Financial Reporting Studio. You can encrypt these communication channels.

Encrypting Financial Reporting RMI communication involves these procedures:

- "Importing Certificate into Keystore" on page 73
- "Configuring RMI Servers" on page 73
- "Configuring Financial Reporting Studio and Financial Reporting Web Application" on page 74

Importing Certificate into Keystore

This discussion assumes that you are using certificates signed by a trusted third-party CA to encrypt Financial Reporting RMI communication. You can use a signed certificate that was obtained for the Financial Reporting host machine that runs services such as web application server and Print Server. See "Required Certificates" on page 31.

If you are using a tool, for example, keytool, to create a custom keystore and to import certificates, see "Creating a Custom Keystore on WebLogic Server and Importing Certificates" on page 35.

Configuring RMI Servers

You configure RMI servers (applications server and Print Server) by adding the following to the JVM startup parameters (shell script files in UNIX servers) or JVMOption Windows registry entries (Windows servers).

```
-Djavax.net.ssl.keyStore=KEYSTORE_LOCATION
-Djavax.net.ssl.keyStorePassword=PASSWORD
-Djavax.net.ssl.trustStore=TRUSTSTORE_LOCATION
```

Explanation of parameter values:

- `KEYSTORE_LOCATION` is the absolute path of keystore where you installed the signed certificate.
- `PASSWORD` is the password of the keystore where you installed the signed certificate.
- `TRUSTSTORE_LOCATION` is the location of the keystore where you installed the root certificate of the CA that signed the certificate.

The registry location for adding these parameters as string values:

- Application Server: `HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\FinancialReporting0\HyS9FRReports`
- Print Server: `HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\FinancialReporting\FRPrintService`

Configuring Financial Reporting Studio and Financial Reporting Web Application

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

```
-Djavax.net.ssl.trustStore=TRUSTSTORE_LOCATION
```

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

The registry location for adding this parameter for Financial Reporting Studio on a Windows server is `HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\Hyperion Reports\HReports\JVM`. The location for adding JVM parameters for the Financial Reporting web application is `HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\FinancialReporting0\HyS9FRReports`.

Updating Financial Reporting Properties in Shared Services Registry

Modify Financial Reporting web application and Print Server properties in the Shared Services Registry to enable SSL socket factories.

➤ To update Financial Reporting properties:

1. **Execute** `FRConfig.cmd` **(Windows) or** `FRConfig.sh` **located in** *EPM_ORACLE_HOME*`/products/financialreporting/bin`.
2. **ON MBeans tab in JConsole, expand, com.hyperion, and then Financial Reporting.**

3. Select **Attributes.**
4. Locate **RMIClientSocketFactory** and set its value to `javax.rmi.ssl.SslRMIClientSocketFactory`.
5. Locate **RMIServerSocketFactory** and set its value to `javax.rmi.ssl.SslRMIServerSocketFactory`.
6. Restart Financial Reporting.

Troubleshooting

Verify that these JVM parameters are set (in shell script files in UNIX servers) or in registry entries (Windows servers) for Financial Reporting processes to write SSL trace messages to a log file.

```
-Djavax.net.debug=all
SysErrFile=
EPM_ORACLE_INSTANCE/diagnostics/logs/services/PROCESS-syserr.log
SysOutFile=EPM_ORACLE_INSTANCE/diagnostics/logs/services/PROCESS-sysout.log
out.log
```

In the preceeding paths, *PROCESS* is either FRPrintService or HyS9FRReports.

SSL for Essbase

Subtopics

Overview

Essbase supports one-way SSL only, in which the Essbase instance (server and agent) is secured using certificates.

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

Supported Platforms

SSL implementation utilizes Oracle's Network Security library, which is not supported on 32-bit UNIX platforms other than Linux. You cannot SSL-enable Essbase on 32-bit UNIX platforms other than Linux. SSL is supported 32-bit and 64-bit servers running Microsoft Windows operating systems.

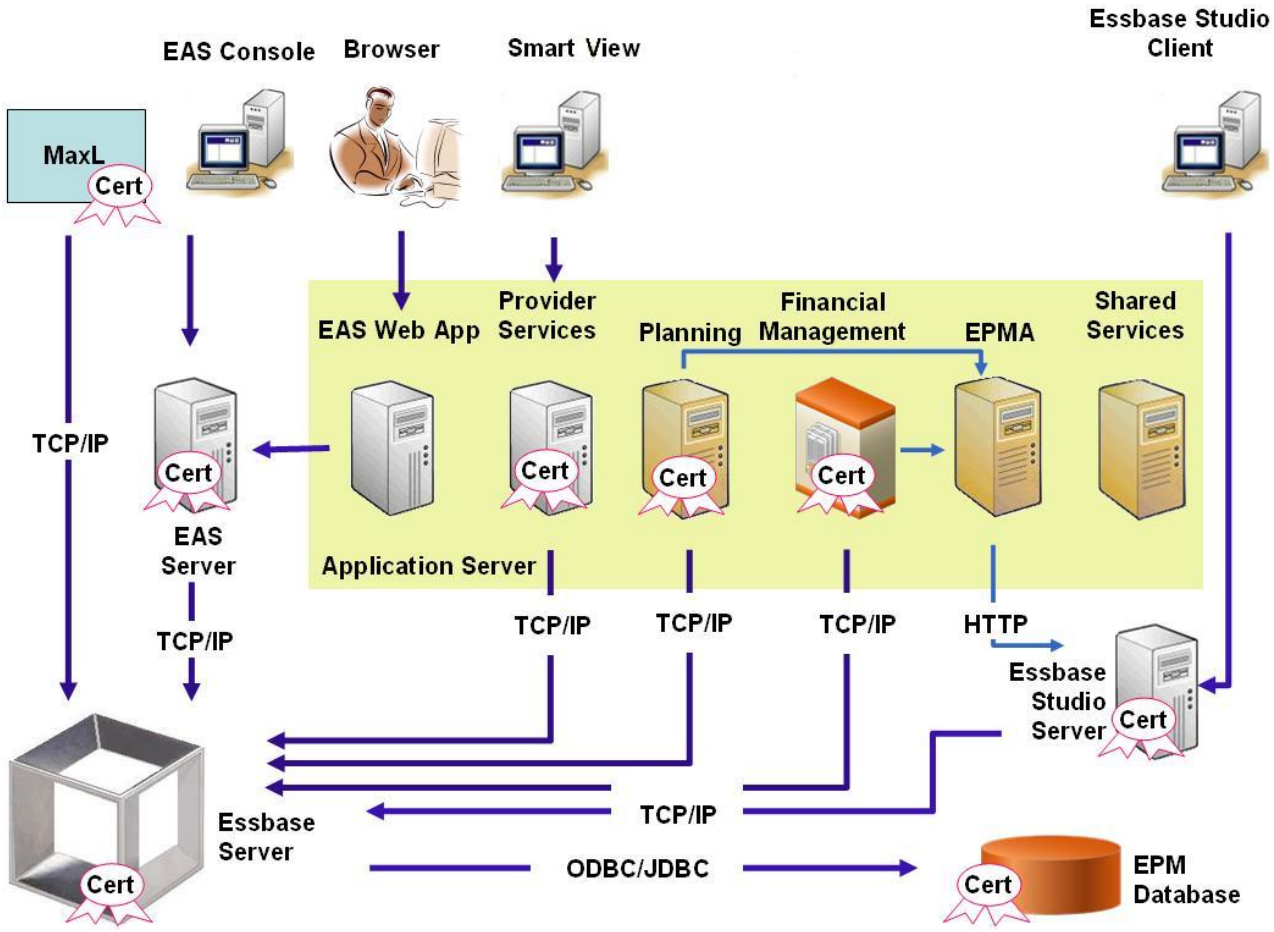
Default Deployment

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

You can control SSL on a per-session basis by specifying the secure protocol and port when you log in. See "Establishing a Per-Session SSL Connection" on page 83.

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

Default deployments of Essbase components in secure mode uses self-signed certificates to enable SSL communication, mainly for testing purposes. Oracle recommends that you use certificates from well-known third-party CAs to SSL-enable Essbase in production environments.



Typically, an Oracle Wallet stores the certificate that enables SSL communication with clients that use Essbase RTC (C APIs) and a Java keystore stores the certificate that enables SSL communication with components that utilize JAPI for communication. To establish SSL communication, Essbase clients and tools store the root certificate of the CA that signed the Essbase Server and Agent certificates. See "Required Certificates and Their Location" on page 77.

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. Use of a wildcard certificate can reduce management time and cost.

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

You require the following certificates:

- A root CA certificate.

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

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

- Signed certificate for Essbase Server and Essbase Agent.

Table 13 Required Certificates and Their Location

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
EPM 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)	• Oracle Wallet • Java Keystore	• root CA certificate • Signed certificate for Essbase Server and Agent
Shared Services Repository		

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

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

Essbase and SSL-Enabled EPM System

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

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

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

Installing and Deploying Essbase Components

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

The Oracle Hyperion Enterprise Performance Management System Installer, Fusion Edition installs an Oracle Wallet and self-signed certificate within *ARBOR_PATH* on the machine that hosts the Essbase instance. In single host deployments, all Essbase components share this certificate.

Using Trusted Third-Party CA Certificates for Essbase

Subtopics

Creating Certificate Requests and Obtaining Certificates

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

You use a tool such as keytool or Oracle Wallet Manager to create a certificate request. For detailed information on creating a certificate request, see the documentation of the tool 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 attests to the validity of the certificate that is used to support SSL. it contains the public key against which the private key that was used to sign the certificate is matched to verify the certificate. You can obtain the root CA certificate from the certificate authority that signed your SSL certificates.

Install the root certificate of the CA that signed the Essbase Server certificate on clients that connect to the Essbase Server or Agent. Ensure that the root certificate is installed in the keystore appropriate for the client. See "Required Certificates and Their Location" on page 77.

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

Oracle Wallet

Refer to Table 13, "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 root CA certificate.

Java Keystore

Refer to Table 13, "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 you are using for detailed instructions. If you are using keytool, use a command, such as the following to import the root certificate:

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

Installing Signed Certificates

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

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

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

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

Updating Default Settings

Subtopics

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

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

Updating Essbase SSL Settings

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

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

➤ To update `essbase.cfg`:

1 **Using a text editor, open** *EPM_ORACLE_INSTANCE*`/EssbaseServer/essbaseserver1/bin/essbase.cfg`.

2 **Enter settings as needed. See Table 14.**

Table 14 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:** `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`

Setting	Description[1]
SSLCipherSuites	A list of cipher suites, in order of preference, to use for SSL communication. See "Available Cipher Suites for Components that Use Essbase C APIs" on page 83. 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_SHA, SSL_RSA_WITH_DES_CBC_SHA
AgentSecurePort	The secure port at which the agent listens. **Default:** 6423 **Example:** AgentSecurePort 16001
WalletPath	Location of the Oracle Wallet (fewer than 1024 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" on page 83. **Default:** CLEAR **Example:** ClientPreferredMode SECURE

[1]The default value is enforced if the property is missing from essbase.cfg.

[2]Essbase becomes inoperational if EnableClearMode and EnableSecureMode are set to FALSE.

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

3 **Save and close** essbase.cfg.

Customizing SSL Properties of JAPI Clients

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

➤ To update SSL properties of JAPI clients:

1 **Using a text editor, open** *EPM_ORACLE_INSTANCE*/EssbaseServer/essbaseserver1/bin/essbase.properties.

2 **Updates the properties as needed. See Table 15 for description of customizable JAPI client properties.**

Table 15 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

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. See "Available Cipher Suites for Components that Use Essbase C APIs" on page 83. **Example**: `SSL_RSA_WITH_AES_256_CBC_SHA,SSL_RSA_WITH_AES_128_CBC_SHA`
`olap.server.ssl.trustManagerClass`	The TrustManager class to use to validate SSL certificate by verifying the signature and checking certificate expiration date. By default, this property is not set to enforce all verification checks. To not enforce verification checks, set the value of this parameter to `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`
`olap.server.ssl.keyManagerClass`	This parameter is not used in this release.

3 **Save and close** `essbase.properties`.

4 **Restart all Essbase components.**

Available Cipher Suites for Components that Use Essbase C APIs

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

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

Establishing a Per-Session SSL Connection

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

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

login admin welcome1 on hostA:secure
```

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

3 Enabling SSO with Security Agents

In This Chapter

Supported SSO Methods

Subtopics

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

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

- "HTTP Header" on page 86
- "Custom Login Class" on page 86
- "HTTP Authorization Header" on page 87
- "Get Remote User from HTTP Request" on page 87

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

HTTP Header

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

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

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

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

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

Custom Login Class

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

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

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

`CSSSecurityAgentIF` comprises two methods: `getUserName` and `getPassword`.

getUserName Method

This method returns the user name for authentication.

```
java.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.

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

Single Sign-on from Oracle Access Manager

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

You can use any header name that provides the value of login attribute to EPM System. You use the header name while configuring Shared Services for SSO from Oracle Access Manager.

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

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

- Configured an HTTP header to pass login attribute value to EPM System.
- Protected the EPM System resources listed in "Resources to Protect" on page 95. Requests to access protected resources are challenged by Oracle Access Manager.
- Unprotected the EPM System resources listed in "Resources to Unprotect" on page 96. 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 Hyperion Enterprise Performance Management System User and Role Security Guide*.**

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

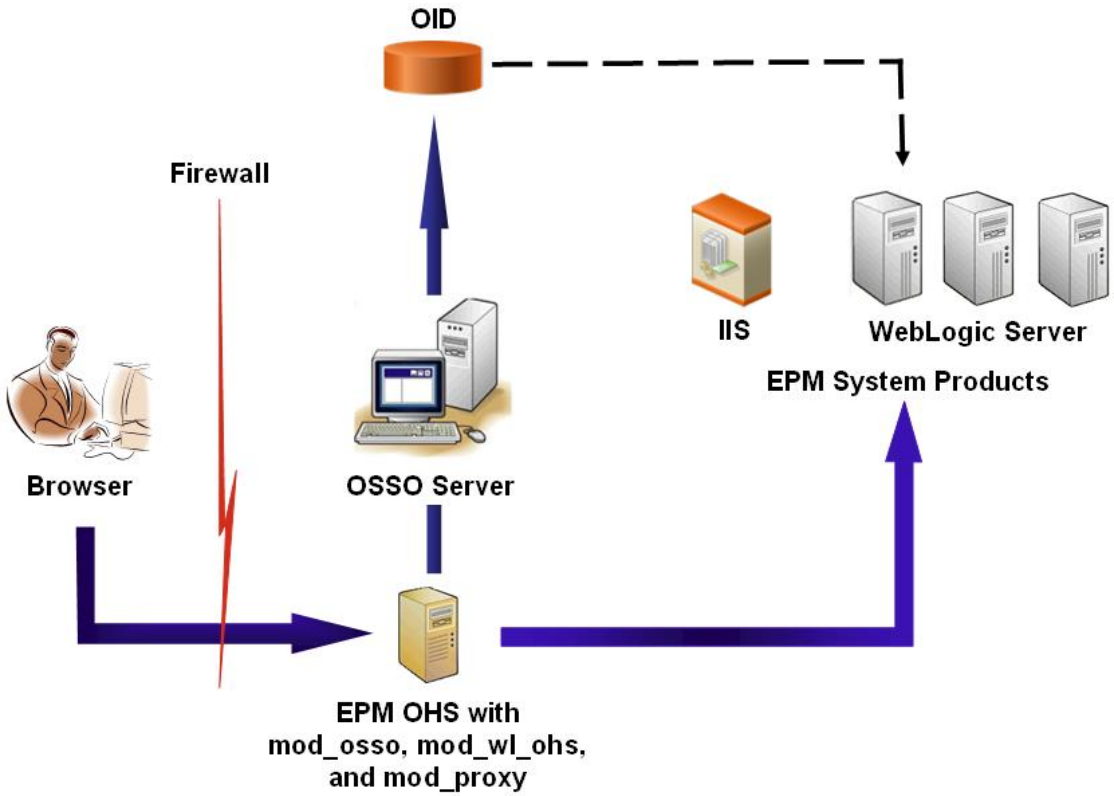
2 **Configure EPM System for SSO. See "Configuring the EPM System for SSO" on page 111.**

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.

OracleAS Single Sign-on

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

Process Flow



The OSSO process:

1. Using an EPM System URL, for example, `http://`*`OSSO_OHS_Server_NAME`*`:`*`OSSO_OHS_Server_PORT`*`/interop/index.jsp`, users access an EPM System component that is defined as an OSSO protected application.
2. Because the URL is under OSSO protection, `mod_osso`, deployed on Oracle HTTP Server, intercepts the request. Using `mod_osso`, Oracle HTTP Server checks for a valid cookie. If a valid cookie is not available in the request, Oracle HTTP Server redirects users to the OSSO Server, which challenges users for credentials, which it authenticates against OID.
3. OSSO Server creates the obSSOCookie and returns control to the `mod_osso` module on the Oracle HTTP Server, which sets the obSSOCookie in the browser. It also redirects the request to the EPM System resource through `mod_wl_ohs` (WebLogic Server) or `mod_proxy` (IIS Server). Before forwarding the request to an EPM System resource, Oracle HTTP Server sets the `Proxy-Remote-User` header which EPM System security uses to enable SSO.
4. The EPM System component verifies that the user whose identity is retrieved from `Proxy-Remote-User` is present in OID. For this process to work, the OID that is configured with the OSSO Server should be configured as an external user directory in Shared Services.

Prerequisites

1. A fully functional Oracle Application Server Infrastructure.

To establish an Oracle Application Server Infrastructure, install and configure Oracle Identity Management Infrastructure 10.1.4. Ensure that OSSO is enabled. Oracle Identity Management Infrastructure 10.1.4 installation includes the following components to support OSSO.

- Oracle 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 front-end to the OSSO Server. This installation includes `mod_osso` that allows you to define partner applications for OSSO.

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

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

2. A fully functional EPM System deployment.

   When you configure the web server for EPM System components, EPM System Configurator configures the following on the Oracle HTTP Server to proxy requests to the application server:

   - `mod_wl_ohs.conf` to proxy requests to WebLogic Server
   - `mod_proxy` to proxy requests to IIS Server

Enabling OSSO for EPM System

Subtopics

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

Registering EPM System Web Server as a Partner Application

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

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

➤ To register EPM System web server as a partner application:

1 **Open a console on the server that hosts the Oracle HTTP Server that front-ends the OSSO Server and navigate to *ORACLE_HOME*`/sso/bin` directory of Oracle HTTP Server, for example to `C:/OraHome_1/sso/bin` (Windows).**

2 **Execute a command similar to the following with `-remote_midtier` option:**

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

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

- `-site_name` identifies the web site of the partner application; for example, `epm.myCompany.com`.
- `-mod_osso_url` indicates the partner application URL, in *`PROTOCOL://HOST_NAME:PORT`* format. This is the URL at which the EPM System web server accepts incoming client requests, for example, `http://epm.myCompany.com:19000`.
- `-config_mod_osso` identifies that the partner application uses `mod_osso`. You must include the `config_mod_osso` parameter to generate `osso.conf`.
- `-update_mode` indicates the update mode. Use `CREATE`, the default, to generate a new record.
- `-remote_midtier` specifies that the `mod_osso` partner application is at a remote mid-tier. Use this option when the partner application is at a different *`ORACLE_HOME`* than that of the OSSO Server.
- `-virtualhost` indicates that the partner application URL is a virtual host. Do not use this parameter if you are not using a virtual host.

  If you are registering a partner application URL tied to a virtual host, you must define the virtual host in `httpd.conf`. See "Optional: Defining Virtual Host" on page 92.
- `-config_file` indicates the path where `osso.conf` file is to be generated.

Optional: Defining Virtual Host

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

➤ To define a virtual host:

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

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

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

Creating `mod_osso.conf`

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

➤ To create `mod_osso.conf`:

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

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

```
LoadModule osso_module C:/Oracle/Middleware/ohs/ohs/modules/mod_osso.so
<IfModule mod_osso.c>
   OssoIpCheck off
   OssoIdleTimeout off
   OssoSecureCookies off
   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" on page 91) creates an obfuscated `osso.conf` in the location identified by the `-config_file` directive.

➤ To relocate `osso.conf`:

1 **Locate the `osso.conf` that was created when you registered EPM System web server as a partner application (see "Registering EPM System Web Server as a Partner Application" on page 91.**

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" on page 92).**

Adding Cache Management Configuration for Reporting and Analysis

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

➤ To add cache management configuration settings:

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

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

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

3 **Save and close `httpd.conf`.**

Configuring EPM System for OSSO

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

➤ To configure EPM System for OSSO:

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

2 **Enable SSO in the EPM System. "Configuring the EPM System for SSO" on page 111**

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

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

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

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

Optional: Enabling Debug Messages on OSSO Server

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

➤ To record debug messages:

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

2 **Set the value of `debugLevel` property to DEBUG.**

```
debugLevel = DEBUG
```

3 **Save and close `policy.properties`.**

Optional: Enabling Debug Messages for Protected Resources

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

➤ To record debug messages for protected resources:

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

2 **Set the value of `OraLogSeverity` property to TRACE.**

```
OraLogSeverity TRACE:32
```

3 **Save and close `httpd.conf`.**

Protecting EPM System Products for SSO

Subtopics

- Resources to Protect
- Resources to Unprotect

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

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

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

Resources to Protect

Table 16 lists the contexts that must be protected. The syntax for protecting a resource (using `interop` as an example) for OSSO:

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

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

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

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

Table 16 EPM System Resources to Protect

EPM System Product	Context to Protect
Shared Services	/interop
Reporting and Analysis Framework	• /raframework • /biplus_webservices
EPM Workspace	/workspace
Financial Reporting	/hr

EPM System Product	Context to Protect
Web Analysis	/WebAnalysis
Performance Management Architect	• /awb • /hyperion-bpma-server
Planning	/HyperionPlanning
Oracle Hyperion Performance Scorecard, Fusion Edition	• /HPSWebReports • /HPSAlerter
Strategic Finance	/HSFWebServices
Oracle Integrated Operational Planning, Fusion Edition	/interlace
Financial Management	• /hfm • /hfmofficeprovider • /hfmlcmserver • /hfmsmartviewprovider • /hfmapplicationservice • /hfmlcmservice
Administration Services	/hbrlauncher
FDM	/HyperionFDM
Calculation Manager	/calcmgr
Oracle Hyperion Provider Services	/aps
Profitability and Cost Management	/profitability
Oracle Hyperion Financial Close Management	/fcc
Disclosure Management[1]	/mappingtool
ERP Integrator	/aif

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

Resources to Unprotect

Table 17 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 17 EPM System Resources to Unprotect

EPM System Product	Contexts to Unprotect
Shared Services	• /interop/framework(.*) • /interop/Audit(.*) • /interop/taskflow* • /interop/WorkflowEngine/* • /interop/TaskReceiver • /framework/lcm/HSSMigration
Performance Management Architect	• /awb/ces.executeAction.do • /awb/lcm.executeAction.do • /awb/appmanager.deployStatusUpdate.do • /awb/jobstask.updateJobStatus.do
EPM Workspace	/workspace/browse/listXML*
Planning	/HyperionPlanning/Smartview
[1]Oracle's Hyperion Reporting and Analysis Framework[2]	• /raframework/browse/listXML • /raframework/wsrp4j(.*) • /raframework/ResourceProxy(.*)
Oracle's Hyperion® Web Analysis*	• /WebAnalysis/wsrp4j(.*) • /WebAnalysis/ResourceProxy(.*)
Oracle Hyperion Financial Reporting, Fusion Edition*	• /hr/common/HRLogon.jsp • /hr/wsrp4j(.*) • /hr/ResourceProxy(.*) • /hr/services/* • /hr/modules/com/hyperion/reporting/web/reportViewer/HRStaticReport.jsp
Hyperion Calculation Manager	/calcmgr/common.performAction.do (for Performance Management Architect)
Oracle Essbase Administration Services	• /eas • /easconsole • /easdocs
Financial Management	/hfm/EIE/EIEListener.asp (for Performance Management Architect)
Planning	• /HyperionPlanning/servlet/HspLCMServlet • /HyperionPlanning/servlet/HspAppManagerServlet (for Performance Management Architect)
Oracle Hyperion Performance Scorecard, Fusion Edition	• /HPSWebReports/wsrp4j(.*) • /HPSWebReports/ResourceProxy(.*) • /HPSWebReports/action/lcmCallBack
Performance Management Architect Data Synchronization	/DataSync/services*

EPM System Product	Contexts to Unprotect
Oracle Hyperion Strategic Finance, Fusion Edition	• /HSFWebServices/HSFWebService.asmx • /HSFWebServices/HSFEntityWebService.asmx
Oracle Integrated Operational Planning, Fusion Edition	• /interlace/services/(.*) • /interlace/anteros/(.*) • /interlace/interlace/(.*) • /interlace/WebHelp/(.*) • /interlace/html/(.*) • /interlace/email-book/(.*)
Profitability and Cost Management	• /profitability/cesagent • /profitability/lcm • /profitability/control • /profitability/ApplicationListener
Oracle Hyperion Financial Data Quality Management ERP Integration Adapter for Oracle Applications	• /aif/services/FDMRuleService • /aif/services/RuleService
Oracle Hyperion Disclosure Management	• /discmanwebservices • /mappingtool/MappingToolWS

[1]When a security agent is enabled for EPM Workspace, Web Analysis, or Financial Reporting that uses SAP Portal, unprotect only the `wsrp4j` URLs (`/raframework/wsrp4j`, `/WebAnalysis/wsrp4j`, and `/hr/wsrp4j`). Also remove the line `com.hyperion.portlet.sso.filter.SMAuthHandler` from `\WEB-INF\classes\auth-handlers.config` in the web application deployments of Reporting and Analysis Framework, Web Analysis, and Financial Reporting. In this scenario, portlets authentication is done using the SAP token. For other portals, protect `wsrp4j` URLs in the security agent.

[2]If you are using Oracle Web Center or Oracle Portal for Portlets, Oracle recommends that you use a security agent such as Oracle Access Manager to protect the system. In this scenario, you must also protect `wsrp4j` URLs using the security agent.

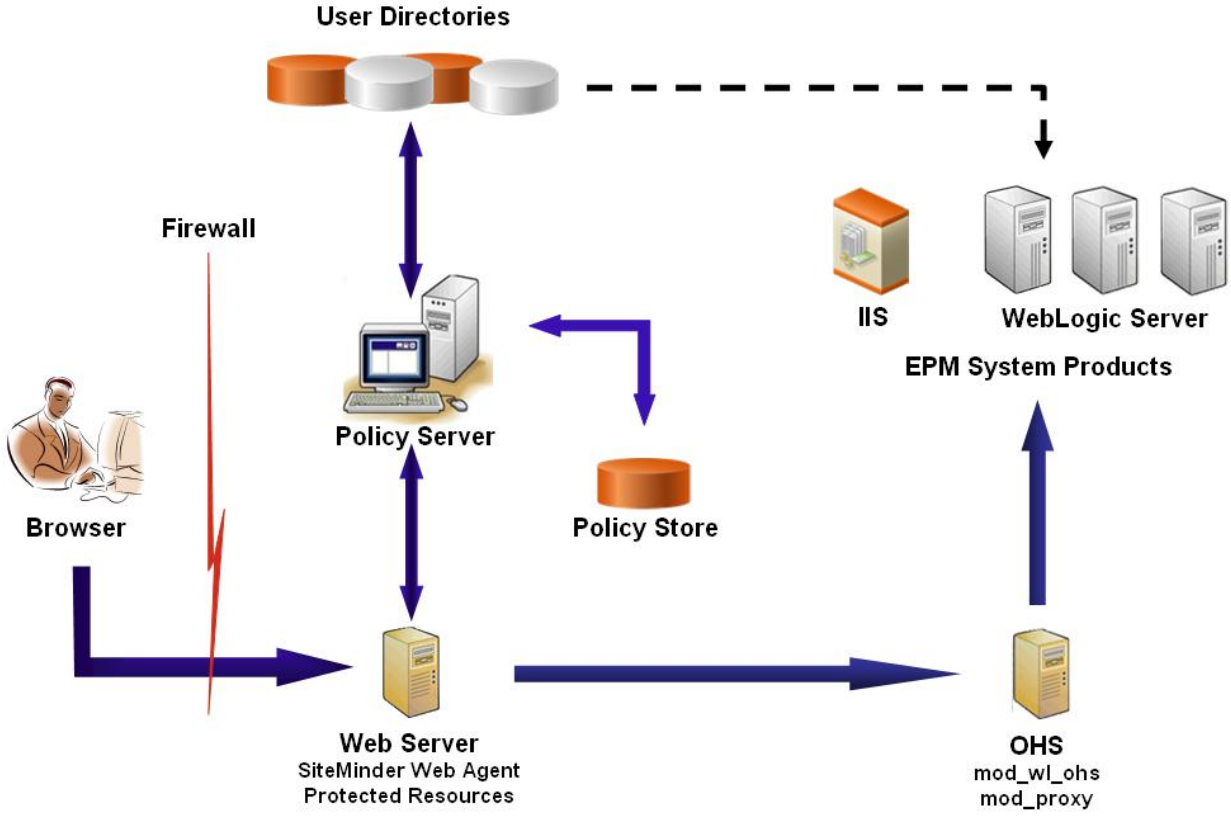
SiteMinder SSO

Subtopics

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.

Process Flow

Illustrated overview of SiteMinder-enabled SSO:



The SiteMinder SSO process:

1. Users try to access a SiteMinder protected EPM System resource. They use a URL that connects them to the web server that front-ends the SiteMinder policy server; for example, `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 WebLogic Server or IIS Server.
4. The EPM System component checks provisioning information and serves up content. For this process to work, the user directories that SiteMinder uses to authenticate users must be configured as external user directories in the EPM System. These directories must be configured as trusted.

Special Considerations

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

Prerequisites

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

   When you configure the web server for EPM System components, EPM System Configurator configures the following on the Oracle HTTP Server to proxy requests to the application server:

   - `mod_wl_ohs.conf` to proxy requests to WebLogic Server
   - `mod_proxy` to proxy requests to IIS

Enabling SiteMinder Web Agent

The web agent is installed on a web server that intercepts requests for EPM System resources. Attempts by unauthenticated users to access a protected EPM System resources forces the web agent to challenge users for SSO credentials. When a user is authenticated, the policy server adds the login name of the authenticated user, which is carried by the header. Thereafter, the HTTP request is passed to the EPM System web server, which redirects the requests. EPM System components extracts the authenticated user credentials from headers.

SiteMinder supports SSO across EPM System products running on heterogeneous web server platforms. If EPM System products use different web servers, you must ensure that the SiteMinder cookie can be passed among web servers within the same domain. You do so by specifying the appropriate EPM System application domain as the value of the `Cookiedomain` property in the `WebAgent.conf` file of each web server.

See the "Configuring Web Agents" in the *Netegrity SiteMinder Agent Guide*.

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

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

➤ To enable SiteMinder Web Agent:

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

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

```
enableWebAgent="YES"
```

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

Configuring the SiteMinder Policy Server

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

The configuration process involves:

- Creating a SiteMinder Web Agent and adding configuration objects appropriate for the SiteMinder web server
- Creating a realm for each EPM System resource that should be protected and adding the web agent to the realm. See "Resources to Protect" on page 95
- Within the realm that was created for protected EPM System resources, create realms for unprotected resources. See "Resources to Unprotect" on page 96
- 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 Hyperion Enterprise Performance Management System User and Role Security Guide* for a brief description of `Login Attribute`.
- Creating rules within the realms with Get, Post, and Put as web agent actions
- Creating a response attribute with `hyplogin=<%userattr="SM_USERLOGINNAME"%>` as the value
- Creating a policy, assigning user directory access, and adding rules that you created for EPM System to Current Members list
- Setting responses for the rules you created for EPM System components

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

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

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

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

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

Enabling SiteMinder in EPM System

Integration with SiteMinder requires that you enable SiteMinder authentication for EPM System products. See "Configuring the EPM System for SSO" on page 111.

Kerberos Single Sign-on

Subtopics

Overview

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

Kerberos is a trusted authentication service, where each Kerberos client trusts the identities of other Kerberos clients (users, network services, and so on) to be valid.

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

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

Support Limitations

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

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

Assumptions: Kerberos Environment

This document assumes the following:

- A fully functional Kerberos-enabled network environment
  - The corporate Active Directory is configured for Kerberos authentication.
  - The application server and HTTP server machines that host EPM System products are within the Kerberos realm.
  - The machines from which EPM System products are accessed are part of the Kerberos realm.
  - Browsers used to access EPM System products are configured for Integrated Windows Authentication. For information on enabling Integrated Windows Authentications, see:

    Internet Explorer documentation on the Microsoft Help and Support web site.

    Firefox documentation on the Firefox Support web site.
- EPM System product users have Kerberos credentials that enable them to log in to client machines in the domain.
- Integrated Windows Authentication is disabled in IIS if it is used as the web server for EPM System products.

Kerberos SSO with WebLogic Server

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

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

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

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

Assumptions

See "Assumptions: Kerberos Environment" on page 103 for assumptions related to the network environment.

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

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

  See Microsoft documentation for detailed information.

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

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

WebLogic Server Procedures to Support Kerberos Authentication

A WebLogic Server administrator should complete these tasks to support Kerberos authentication:

- Configure the WebLogic domain of EPM System. See "Configuring EPM System WebLogic Domain" on page 105.
- Create an authentication provider. See "Creating an LDAP Authentication Provider in WebLogic Server" on page 105.
- Create a Negotiate Identity Asserter. See "Creating a Negotiate Identity Asserter" on page 105.
- Create a Kerberos identification. See "Creating Kerberos Identification for WebLogic Server" on page 106.
- Create a Kerberos configuration file. See "Creating Kerberos Configuration File" on page 106.
- Update WebLogic startup script. See "Updating WebLogic Startup Script" on page 106.

- Configure authorization policies. See "Configuring Authorization Policies" on page 107.
- Deploy and use SSODiag to verify that the WebLogic Server is ready to support Kerberos SSO for EPM System. See "Using SSODiag to Test the Kerberos Environment" on page 107.

Configuring EPM System WebLogic Domain

Generally, EPM System products are deployed into `epmsystem1`, which is the default WebLogic domain. This domain is identified also as *`EPM_ORACLE_INSTANCE`*.

➤ To configure the EPM System WebLogic domain for Kerberos authentication:

1 **Install EPM System components.**

2 **Create the WebLogic domain by configuring and deploying Foundation Services only.**

3 **Configure the WebLogic domain from the preceding step for Kerberos authentication. See "Configuring Single Sign-on with Microsoft Clients" in *Oracle Fusion Middleware Securing Oracle WebLogic Server* guide.**

Steps involved:

- "Creating an LDAP Authentication Provider in WebLogic Server" on page 105
- "Creating a Negotiate Identity Asserter" on page 105
- "Creating Kerberos Identification for WebLogic Server" on page 106
- "Updating WebLogic Startup Script" on page 106
- "Configuring Authorization Policies" on page 107

Creating an LDAP Authentication Provider in WebLogic Server

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

Creating a Negotiate Identity Asserter

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

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

Creating Kerberos Identification for WebLogic Server

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

➤ To create Kerberos identification for WebLogic Server:

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

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

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

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

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

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

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

Creating Kerberos Configuration File

Kerberos configuration properties are defined in `kerb5.ini`. This configuration file is required to use Kerberos administration tools such as `kinit` and `ktab`.

See Configuring Your Network Domain to Use Kerberos in *Oracle Fusion Middleware Securing Oracle WebLogic Server 11g Release 1 (10.3.1)*.

Updating WebLogic Startup Script

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

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

➤ To update JVM Startup options in Windows registry:

1 **Open Windows Registry Editor.**

2 Find the Foundation Services key by selecting **My Computer**, then **HKEY_LOCAL_MACHINE**, then **Software**, then **Oracle**, and then **Foundation Services**.

3 Add the following string values:

Table 18 JVM Startup Options for Kerberos Authentication

Name	Type	Data
JVMOption14	REG_SZ	`-Djava.security.krb5.kdc=`*`Active Directory host name or IP address`*
JVMOption16	REG_SZ	`-Djava.security.auth.login.config=krb5Login.conf`
JVMOption16	REG_SZ	`-Djavax.security.auth.useSubjectCredsOnly=false`
JVMOption17	REG_SZ	`-Djava.security.enableNegotiate=true`

Configuring Authorization Policies

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

For sample policy configuration steps, see "Creating Policies for SSODiag" on page 108.

Using SSODiag to Test the Kerberos Environment

Subtopics

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

Deploying SSODiag

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

➤ To deploy an configure SSOdiag:

1 Log on to the WebLogic Server Administration Console for EPM System domain.

2 Using the **Install Application Assistant**, select *`EPM_ORACLE_HOME`*`/products/Foundation/AppServer/InstallableApps/common/SSODiag.war` as the web application to install.

3 Deploy SSODiag as an application (choose **Install this deployment as an application** as targeting style).

4 **Activate the changes you made.**

Configuring Oracle HTTP Server for SSODiag

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

➤ To configure URL forwarding in Oracle HTTP Server:

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

2 **Add a** `LocationMatch` **definition for SSODiag:**

```
<LocationMatch /SSODiag/>
    SetHandler weblogic-handler
    WeblogicCluster myServer:28080
</LocationMatch>
```

In the preceding sample, `myServer` denotes the Foundation Services host machine and `28080` represents the port at which Shared Services listens for requests.

3 **Save and close** `mod_wl_ohs.conf`.

4 **Restart Oracle HTTP Server.**

Creating Policies for SSODiag

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

`http://`*OHS_HOST_NAME*`:`*PORT*`/SSODiag/krbssodiag`

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

➤ To create policies to protect SSODiag:

1 **In the Change Center in WebLogic Server Administration Console for EPM System domain, select Lock & Edit.**

2 **Select Deployments, then SSODiag, then Security, then Roles, and then URL Patterns.**

3 **Create the following URL patterns:**

- `/`
- `/index.jsp`

4 **Modify each URL pattern that you created:**

a. From the list of URL patterns in **Stand-Alone Web Application URL Patterns**, open the pattern (/) that you created by clicking it.

b. Select **Add Conditions**.

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**.

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 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.
```

Configuring Foundation Services for Kerberos Authentication

Subtopics

Caution! Complete this step only after SSODiag successfully retrieves the Kerberos principal name. See "Using SSODiag to Test WebLogic Server Configuration for Kerberos Authentication" on page 109.

Changing the Security Model

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

➤ To change the security model:

1. **Using a text editor, open** *MIDDLEWARE_HOME*`/user_projects/domains/EPMSystem/config/config.xml`.
2. **Locate the following element in the application deployment descriptor for each Foundation Services component:**

   ```
   <security-dd-model>DDOnly</security-dd-model>
   ```

3. **Change the security model as follows for each component:**

   ```
   <security-dd-model>CustomRolesAndPolicies</security-dd-model>
   ```

4. **Save and close** `config.xml`.

Updating EPM System Security Configuration

Change EPM System security configuration to enable Kerberos SSO.

➤ To configure EPM System for Kerberos authentication:

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

   In **Security Options**, select the settings in Table 19 to enable Kerberos SSO.

Table 19 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 Other EPM System Components

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

Change the security model for each EPM System Component deployed into the WebLogic domain where Foundation Services is deployed. See "Changing the Security Model" on page 110.

Configuring the EPM System for SSO

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

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

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

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

➤ To enable SSO from a web identity management solution:

1. **Launch the Shared Services Console. See "Launching Shared Services Console" in the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*. Log in as a Shared Services Administrator.**
2. **Select Administration, then Configure User Directories.**
3. **Verify that the user directories used by the web identity management solution are configured as external user directories in Shared Services.**

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

   See "Configuring User Directories" in the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.
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 Table 20. See "Supported SSO Methods" on page 85.

      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 20 Preferred SSO Mechanisms for Web Identity Management Solutions

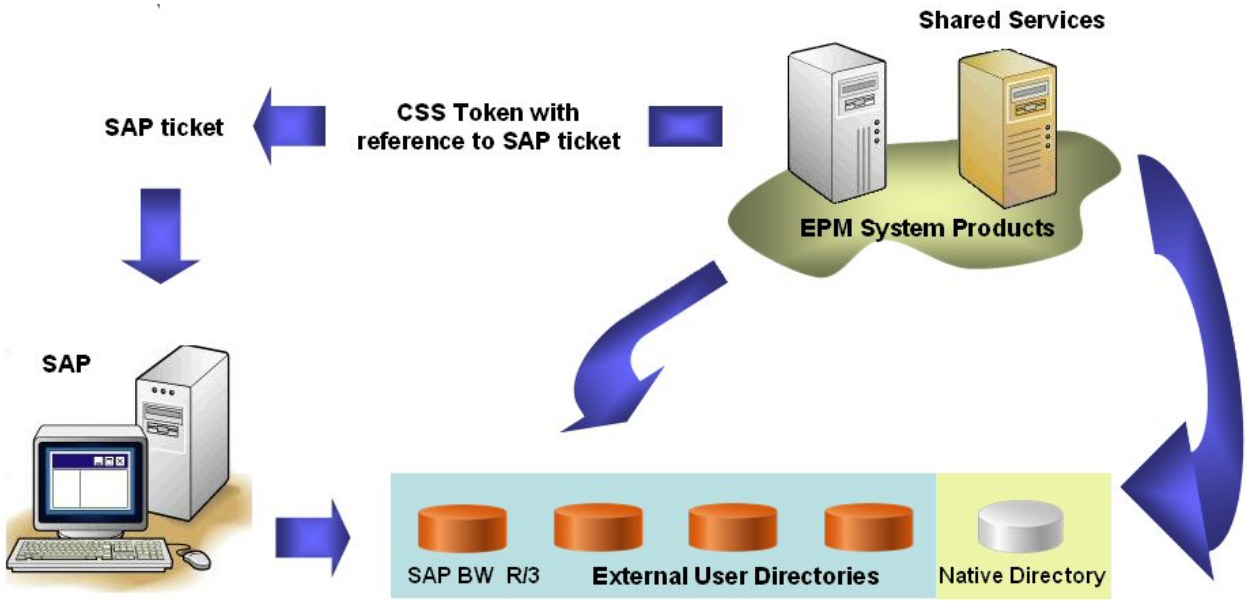
Web Identity Management Solution	Recommended SSO Mechanism
Oracle Access Manager	`Custom HTTP Header`[1]
OSSO	Select `Other` in SSO **Provider or Agent** and `Custom HTTP Header` in **SSO Mechanism**. Enter `Proxy-Remote-User` as the name of the custom HTTP header.
SiteMinder	`Custom HTTP Header`
Kerberos	WebLogic Server: `Custom HTTP Header`

[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 with SAP Enterprise Portal

EPM System products handle SSO to SAP Enterprise Portal by issuing an SAP logon ticket. This action enables users who log in to EPM System products to navigate seamlessly to SAP applications. The illustrated concept:

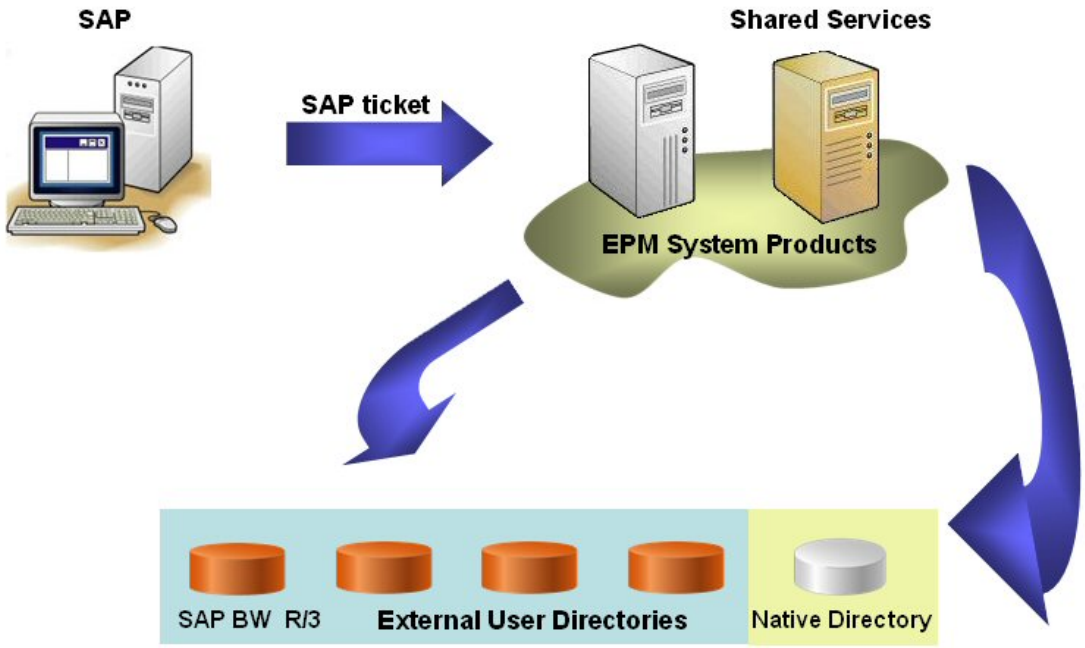


When a user logs in, the EPM System product authenticates the user against configured user directories, including Native Directory, and issues an EPM System logon token. This token enables SSO to EPM System products. It also generates a SAP logon ticket if the user is defined in the SAP provider.

Note: For SSO with SAP to work, you must configure SAP native repository as an external user directory on Shared Services.

When the user subsequently navigates to the SAP system or uses an SAP data source, the SAP logon ticket contained in the EPM System token is passed to SAP to enable SSO. The SAP system assumes the responsibility to validate the credentials in the SAP logon ticket.

EPM System products handle SSO from SAP Enterprise Portal by accepting an SAP logon ticket. This action enables users who log in to SAP Enterprise Portal to navigate seamlessly between SAP and EPM System products. The illustrated concept:



When a user logs in to SAP Enterprise Portal, SAP authenticates the user.

When the user navigates to an EPM System product, the SAP ticket is passed to the EPM System product. Using an SAP certificate stored on the Shared Services server machine, the EPM System retrieves the user name, which is trusted as being that of a valid user. The EPM System product queries user directories to determine the user's groups. Using the group information, EPM System product gets provisioning information.

Note: The SAP provider must be configured as a user directory in Shared Services for this process to work.

Nested SAP Groups

After configuring an SAP user directory, available SAP users and groups are displayed in Shared Services Console. Shared Services considers the SAP roles to be the equivalents of groups created by any corporate directory server. Each role from the SAP user directory is displayed as a distinct group in Shared Services Console. Shared Services, however, does not retrieve the relationships between simple and composite roles within the SAP user directory. If needed, you can create nested groups in Native Directory to mimic the relationship that existed between the simple and composite roles in the SAP user directory. This approach, however, has performance implications and should be avoided if possible.

Prerequisites

- All SAP systems within the SAP landscape must be set up for SSO with the SAP logon ticket. User names must be normalized across the SAP landscape so that a user name in one SAP system refers to the same user across all SAP systems. See SAP documentation for more information.

- Copy or download the SAP JCo binaries or shared libraries into *EPM_ORACLE_HOME*/common/SAP/bin directory.

  JCo binaries and shared libraries are available in your SAP distribution. Registered SAP users can download them from the SAP web site https://service.sap.com/connectors.

- Copy or download the SAP JCo archives (JAR files) and libraries into *EPM_ORACLE_HOME*/common/SAP/lib directory.

  JCo archives and libraries are available in your SAP distribution. Registered SAP users can download them from the SAP web site https://service.sap.com/connectors.

  The following libraries are required to verify that the SAP SSO ticket provided to EPM System products. This step is required only if EPM System products are plugged into SAP Enterprise Portal.

  - com.sap.security.core.jar
  - com.sap.security.api.jar
  - sapjco.jar
  - sap.logging.jar
  - iaik_jce.jar
  - iaik_jce_export.jar (if using the export version of the IAIK-JCE libraries)

- Expand the contents of each SAP JAR file by running explodejar.sh or explodejar.bat, available in *EPM_ORACLE_HOME*/common/SAP/lib directory.
- Install the SAP Digital Certificate (SAP X509 certificate, SAP.keystore) in a convenient location.
- Using Shared Services Console, configure SAP provider as an External user directory. See "Configuring SAP R3 Native Directory" in the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.
- Using Shared Services Console, provision SAP users and groups to provide them access to EPM System products. See "Provisioning Users and Groups" in the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Single Sign-on Options for Smart View

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

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

4 Using a Custom Authentication Module

In This Chapter

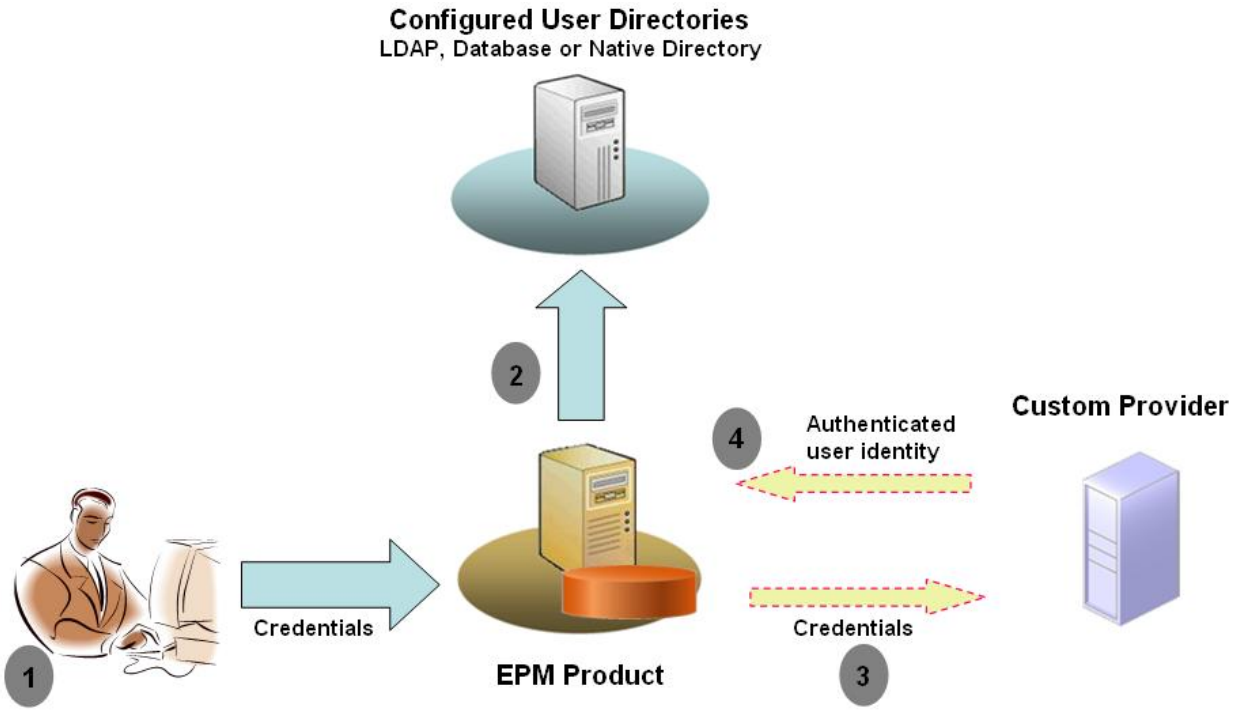
Overview

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

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

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

The following illustration presents a sample custom authentication scenario:



For example, you can use RSA SecurID infrastructure as the custom provider to ensure transparent strong authentication to the EPM System. An overview:

1. The user enters credentials (generally, user name and password) to access an EPM System product. These credentials should uniquely identify the user to the provider used by the custom authentication module. For example, if you are using an RSA SecurID infrastructure to authenticate users, the user enters an RSA user ID and PIN (not an EPM System user ID and password).
2. Using the search order (see "Search Order" on page 120), EPM System cycles through configured user directories to locate the user.
   - If the current user directory is not configured for custom authentication, EPM System tries to locate and authenticate the user through EPM System authentication.
   - If the user directory is configured for custom authentication, EPM System delegates the authentication process to the custom module.
3. If EPM System delegates authentication to the custom module, the custom authentication module accepts the credentials and uses its own logic to direct user authentication against a custom provider, for example, RSA SecurID infrastructure.
4. If the custom authentication module authenticates the user against its provider, it returns the user name to the EPM System, or it returns a Java exception.

   The user name returned by the custom authentication module must be identical to a user name in one of the user directories that is enabled for custom authentication.

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

- If the custom authentication module throws an exception or returns a null user, EPM System continues to search for the user in the remaining user directories in the search order that are not enabled for custom authentication. If a user who matches the credentials is not found, EPM System displays an error.

Use-Case Examples and Limitations

Custom authentication implementation scenarios include the following:

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

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

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

Prerequisites

- 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 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

Subtopics

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

Search Order

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

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

Impact of Custom Authentication on Search Order

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

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

Understanding the Custom Authentication Flow

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

- "Use Case Scenario 1" on page 120
- "Use-case Scenario 2" on page 122
- "Use-case Scenario 3" on page 122

Use Case Scenario 1

Table 21 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 21 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 22 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)
`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

Table 23 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 23 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 24 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

Table 25 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 25 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 26 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" on page 120). 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 a an error message if authentication is not successful. For the authentication process to be completed, the user name returned by the custom authentication module must be present in one of the user directories included in Shared Services search order. EPM System security framework supports the *`username@providerName`* format.

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

See "Sample Code 1" on page 135 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.

Authenticate Method

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

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

Deploying the Custom Authentication Module

Subtopics

Only one custom module is supported for an EPM System deployment. You can enable custom authentication for one or more user directories in the search order.

The custom authentication module must implement the public interface `CSSCustomAuthenticationIF`, defined in the `com.hyperion.css` package. This document assumes that you have a fully functional custom module that defines the logic for authenticating users against the user provider of your choice. After you develop and test a custom authentication module, you must implement it in EPM System environment.

Overview of Steps

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

- Stop EPM System products including Shared Services and any systems that use Shared Services APIs.
- Copy the custom authentication module Java archive `CustomAuth.jar` into *EPM_ORACLE_HOME*`/common/jlib/11.1.2.0`.
- Update user directory settings in Shared Services. See "Updating Settings in Shared Services" on page 125.
- Start Shared Services followed by other EPM System products.
- Test your implementation. See "Testing Your Deployment" on page 126.

Updating Settings in Shared Services

Subtopics

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

Updating User Directory Configurations

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

➤ To update user directory configuration:

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

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

Updating Security Options

Ensure that `CustomAuth.jar` is available in *EPM_ORACLE_HOME*`/common/jlib/11.1.2.0` before starting the following procedure.

➤ To update security options:

1. Log in to Shared Services Console as a Shared Services administrator.
2. Select **Administration**, and then **Configure User Directories**.
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 EPM System search order.

➤ To test custom authentication:

1 **Ensure that EPM System products are running.**

2 **Access an EPM System product; for example, EPM Workspace.**

3 **Log in as a user defined on a user directory for which custom authentication is enabled.**

   a. In **Username**, enter your user identifier; for example, an RSA User ID.

   b. In **Password**, enter a password; for example; an RSA PIN.

   c. Click **Login**.

4 **Verify that you can access EPM System product resources.**

5 Guidelines for Securing EPM System

In This Chapter

Implementing SSL

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

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

Changing the Admin Password

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

Edit the *admin* user account to change the password. See "Modifying User Accounts" in the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Regenerating Encryption Keys

Use the Shared Services Console to periodically regenerate the following:

- Single Sign-On Token

> **Caution!** Taskflows used by Financial Management; Oracle Hyperion EPM Architect, Fusion Edition and Oracle Hyperion Profitability and Cost Management, Fusion Edition 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 "Setting Encryption Options" in *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide* for detailed procedures.

Changing Database Passwords

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

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

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

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

   Shared Services Only:

   > **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 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.

a. From the configuration task list of the product in EPM System Configurator, select **Configure Database**.

b. On the Database Configuration page, select **Perform 1st-time configuration of database**.

c. Specify the new password of the user whose account was used to configure EPM System product database. Do not change any other settings.

d. Click **Next**.

e. Select **Reuse the existing database**.

f. Continue the configuration, and click **Finish** when you are done.

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

4 **Start EPM System products and services.**

Securing Cookies

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

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

Modify the WebLogic Server session descriptor to secure WebLogic Server cookies. Set the value of `cookieSecure` attribute in the `session-param` element to `true`. See http://e-docs.bea.com/wls/docs92/webapp/weblogic_xml.html for detailed information.

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 Hyperion Enterprise Performance Management System User and Role Security Guide*.

Reviewing Security Reports

The Security Report contains audit information related to the security tasks for which auditing is configured. Generate and review this report from Shared Services Console on a regular basis, especially to identify failed login attempts across EPM System products and provisioning changes. Select **Detailed View** as a report generation option to group the report data based on

attributes that were modified and the new attribute values. See "Generating Reports" in the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

Customizing Authentication System for Strong Authentication

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

Turning off Detailed Financial Management Error Messages

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

➤ To hide error messages containing detailed technical information:

1. **On Windows server that hosts Financial Management, launch the Windows Registry Editor.**
2. **Navigate to** `HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\Hyperion Financial Management`
3. **Create a new DWORD value using these settings:**

   Value name: `DisableTechnicalError`

   Value data: `1` (set this to `0` to display detailed messages)
4. **On the Windows server that hosts the IIS Server that hosts Financial Management, launch the Windows Registry Editor.**
5. **Navigate to** `HKEY_LOCAL_MACHINE\SOFTWARE\Hyperion Solutions\Hyperion Financial Management\web`
6. **Create a new DWORD value using these settings:**

   Value name: `DisableAspTechnicalErrorMessage`

   Value data: `1` (set this to `0` to display detailed messages)

Encrypting UDL File (Financial Management)

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

See "Encrypting UDL Files" in *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

Disabling EPM Workspace Debugging Utilities

- For troubleshooting purposes, EPM Workspace ships with uncrunched JavaScript files. For security purposes, you should remove these uncrunched JavaScript files from your production environment:
  - Create a backup copy of *EPM_ORACLE_HOME*/common/epmstatic/wspace/js/ directory.
  - Except for the file *DIRECTORY_NAME*.js, delete the .js files from each subdirectory of *EPM_ORACLE_HOME*/common/epmstatic/wspace/js.

    Each subdirectory contains a .js file that bears the name of the directory. For example, *EPM_ORACLE_HOME*/common/epmstatic/wspace/js/com/hyperion/bpm/web/common contains Common.js. Remove all .js files except the one that bears the name of the directory, in this case; Common.js.
- EPM Workspace provides some debug utilities and test applications, which become accessible if EPM Workspace is deployed in debug mode. For security purposes, administrators should turn off client side debugging in EPM Workspace.

  To turn off debugging mode:

  1. Log in to Oracle Enterprise Performance Management Workspace, Fusion Edition, 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 from 1.5 to 1.6.

A Custom Authentication Sample Code

In This Appendix

Sample Code 1

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;
import org.apache.log4j.Logger; // imports Log4j's Logger
  public class CustomAuthenticationImpl
               implements CSSCustomAuthenticationIF {
    //Get the Logger to log exception or debug information
    //Log information is written to the Shared Services security log
    static Logger logger=Logger.getLogger
                         ("com.hyperion.css.custom.CustomAuthenticationImpl");
      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:
                //ensure debug is enabled using logger.isDebugEnabled()
                //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 EPM System product.
- Password: The password set for the user in the user directory where the custom module authenticates the user. The user enters the value of this parameter while logging into an EPM System product.

Sample Code 2

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

```
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{
  //get the Logger to log Exception or other info useful for debugging
  /*static Logger logger = Logger.getLogger
    ("com.hyperion.css.custom.CSSCustomAuthenticationImpl"); */
  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){

        }
  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 flatfile */

public String authenticate(Map context,String userName, String password)throws
Exception{

  //UserName : user input for the userName
  //Password : user input for password
  //context  : Map, can be used to additional information required by
  //           the custom authentication module.

  String authenticatedUserKey = userName + ":" + password;

  if(users.get(authenticatedUserKey)!=null)
     return(String)users.get(authenticatedUserKey);
  else throw new Exception("Invalid User Credentials");
    }

/* Refer to this authenticate method snippet to return username in
   username@providername format */

public String authenticate(Map context,String userName, String password)throws
Exception{

  //UserName : user input for userName
  //Password : user input for password
  //context  : Map, can be used to additional information required by
  //           the custom authentication module.

  //Your code should uniquely identify the user in a custom provider and in a
```

```
configured
  //user directory in Shared Services. EPM Security expects you to append the provider
  //name to the user name. Provider name must be identical to the name of a custom
  //authentication-enabled user directory specified in Shared Services.

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

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

Data File for Sample Code 2

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

Use the following as the contents of the flat file that is used as the custom user directory to support the custom authentication module implemented by Sample Code 2 (see "Sample Code 2" on page 136):

```
xyz:password:admin
test1:password:test1@LDAP1
test1:password:test1
test1@LDAP1:password:test1@LDAP1
test1@1:password:test1
user1:Password2:user1@NTLM1
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

In This Appendix

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

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

Custom Login Class Sample Code

This sample code illustrates the implementation of the default `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 java.util.HashMap;
import java.util.Locale;
import java.util.Map;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * X509CertificateAuthImpl implements the CSSSecurityAgentIF interface It accepts
 * the X509 certificate of the authenticated user from the Web Server via a
 * header, parses the certificate, extracts the DN of the User and
```

```
 * authenticates the user.
 */
public class X509CertificateSecurityAgentImpl implements CSSSecurityAgentIF
{
    static final String IDENTITY_ATTR = "CN";
    String g_userDN = null;
    String g_userName = null;
    String hostAdrress= null;
    /**
     * Returns the User name (login name) of the authenticated user,
     * for example demouser. See CSS API documentation for more information
     */
    public String getUserName(HttpServletRequest req, HttpServletResponse res)
            throws Exception
    {
        hostAdrress = req.getServerName();
        String certStr = getCertificate(req);

        String sCert = prepareCertificate(certStr);

        /* Authenticate with a CN */
        parseCertificate(sCert);

        /* Authenticate if the Login Attribute is a DN */
        if (g_userName == null)
        {
            throw new Exception("User name not found");
        }
        return g_userName;
    }

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

    /**
     * Get the Certificate sent by the Web Server in the HYPLOGIN header.
     * If you pass a different header 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:

- Create and test the custom login class. See "Custom Login Class Sample Code" on page 139.

  You can use any name for your custom class.
- Package the custom login class into `CustomAuth.jar`
- Copy `CustomAuth.jar` into *`EPM_ORACLE_HOME`*`/common/jlib/11.1.2.0/`.
- Oracle recommends that you enable Client Certificate Authentication if you are using a custom login class. See "Client Certificate Authentication (Two-Way SSL)" on page 67.

C Using the Update Native Directory Utility

In This Appendix

About the Update Native Directory Utility

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

Update Native Directory Utility Installation Location

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

Update Native Directory Utility Options

Update Native Directory Utility creates log files and CSS_MIGRATION_DELETE_LIST.csv. See "Log Files Generated by Update Native Directory Utility" on page 146.

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

Table 27 Update Native Directory Utility Command Line Options

Option	Description
`-noprompt`	**Optional:** Use this option to invoke silent mode operation. Used to schedule jobs involving Update Native Directory Utility. **Example:** `updateNativeDir -noprompt` updates Native Directory in silent mode.
`-delete all`	**Optional:** Use this option to delete all stale Native Directory identities that are marked for deletion.
`-delete` *`PATH_OF_DELETE_LIST`*	**Optional:** Use this option to delete stale Native Directory identities listed in `CSS_MIGRATION_DELETE_LIST.csv`. See "Identifying Stale Data" on page 144.
`-cssLocation`	**Optional:** Use this option to specify the absolute path of the EPM security configuration file. If you do not specify this option, the utility initializes using the security configuration file available in Shared Services Registry.

Using Update Native Directory Utility

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

- "Identifying Stale Data" on page 144
- "Deleting Stale Data" on page 145
- Resolve ambiguous identities. Ambiguous identities are identities that the utility failed to resolve. You must manually resolve these identities.

Updating the Update Native Directory Utility Settings

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

➤ To update utility settings:

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

Identifying Stale Data

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

➤ To identify stale Native Directory data:

1 **Modify Update Native Directory Utility settings. See "Updating the Update Native Directory Utility Settings" on page 144.**

2 **Using a command prompt window or console on the server that hosts an EPM System product, navigate to** *EPM_ORACLE_HOME*`/common/utilities/UpdateNativeDir`**; for example, to** `C:\Oracle\Middleware\EPMSystem11R1\common\utilities\UpdateNativeDir` **on a Windows server.**

3 **Execute a command:**

- `updateNativeDir -cssLocation` *`LOCATION_OF_CSS.xml`* (Windows)
- `updateNativeDir.sh -cssLocation` *`LOCATION_OF_CSS.xml`* (UNIX)

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

4 **Enter** `1` **at the following Update Native Directory Utility query:**

```
Do you want to proceed? [0->No/1->Yes] :
```

Deleting Stale Data

Before deleting stale data, verify the contents of `CSS_MIGRATION_DELETE_LIST.csv`. See "Identifying Stale Data" on page 144.

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

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

➤ To delete stale Native Directory data:

1 **Modify Update Native Directory Utility settings. See "Updating the Update Native Directory Utility Settings" on page 144.**

2 **Using a command prompt window or console on the server that hosts an EPM System product, navigate to** *EPM_ORACLE_HOME*`/common/utilities/UpdateNativeDir`**; for example, to** `C:\Oracle\Middleware\EPMSystem11R1\common\utilities\UpdateNativeDir` **on a Windows server.**

3 **Execute a command. For a list of options you can specify, see "Update Native Directory Utility Options" on page 143.**

Note: You can combine the `-noprompt` and `-cssLocation` directives with these commands.

- `updateNativeDir -delete` *`PATH_OF_DELETE_LIST`*
- `updateNativeDir -delete all`

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

4 **Enter `1` at the following Update Native Directory Utility query:**

```
Do you want to proceed? 0->No/1->Yes] :
```

Log Files Generated by Update Native Directory Utility

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

- `CSSMigration-Ambiguous_`*`time_stamp`*`.log` lists ambiguous identities that Update Native Directory Utility could not resolve. You must manually update the identities listed in this file.
- `CSSMigration-Deleted_`*`time_stamp`*`.log` lists the identities that Update Native Directory Utility deleted from Native Directory.
- `CSSMigration-Updated_`*`time_stamp`*`.log` lists the Native Directory identities that Update Native Directory Utility updated in Native Directory to reflect the changes to the identity in an external user directory.
- `CSSMigration-Ignored_`*`time_stamp`*`.log` lists the entries on which no action was taken because they needed no update.

D

Migrating Users and Groups Across User Directories

In This Appendix

Overview

Organizations may retire their corporate user directory and switch to a different user directory, causing the user and group identities of provisioned EPM System users to become stale. For example, an organization may replace its NTLM user directory with OID or Active Directory. EPM System products become inaccessible if the provisioning information available to EPM System products is not updated to reflect the identity of the users and groups in the new corporate user directory.

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

Prerequisites

- EPM System users and groups whose provisioning data is being migrated across user directories must be available in the target user directory.

  Group relationships that exist in the source user directory must be maintained in the target user directory.
- User names of EPM System users must be identical across source and target user directories.

Migration Procedure

Subtopics

Export Native Directory Data

Use Lifecycle Management to export the following from Native Directory:

- Native Directory Groups
- Assigned roles
- Delegated lists

Lifecycle Management creates multiple export files, generally in *EPM_ORACLE_INSTANCE*/import_export/*USER_NAME*/EXPORT_DIR/resource/Native Directory, where *USER_NAME* is the identity of the user; for example, admin@Native Directory, who performed the export operation and *EXPORT_DIR* is the name of the export directory. Typically, these files are created:

- Groups.csv
- Roles.csv
- Delegated Lists.csv
- Assigned Roles/*PROD_NAME*.csv for each deployed application, where *PROD_NAME* is the name of an EPM System component; for example, Shared Services.

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

➤ To export provisioning data from Native Directory:

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

7 Click **Execute Migration.**

Prepare EPM System for Migration

Subtopics

Optional: Add the Target User Directory as an External User Directory

Add the target user directory as an external user directory if you moved the user accounts from the source user directory to a new user directory. See "Chapter 3, Configuring User Directories" in the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

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

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

While configuring the target user directory as an external user directory, verify that the Login Attribute property points to the attribute whose value was originally used as the user name in the source user directory. See "Prerequisites" on page 147.

Change the Search Order of the Target User Directory

Note: You can delete the source user directory configuration from EPM System or remove the source user directory from the search order instead of changing the search order assignment. See "Removing a Search Order Assignment" in the *Oracle Hyperion Enterprise Performance Management System User and Role Security Guide*.

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

Restart the EPM System

Restart Oracle's Hyperion® Foundation Services and other EPM System components to enforce the changes you made. Restarting Oracle's Hyperion® Shared Services forces the EPM System to renew its cache.

Edit Import Files

You use the export files that Lifecycle Management created as the source for recreating the data in Native Directory. The export files are generated in the directory that you specified while exporting data from Native Directory. See "Export Native Directory Data" on page 148.

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

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

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

➤ To edit an import file:

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

Import Updated Data

Run Lifecycle Management with `create/update` option to import the data you exported earlier from Native Directory. See "Export Native Directory Data" on page 148.

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

➤ To import updated provisioning data into Native Directory:

1. **In the View pane of Oracle's Hyperion® Shared Services Console, expand File System.**
2. **Select the file system location of the import files.**

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

4 Click **Define Migration**.

5 In **Source Options**, click **Next**.

6 In **Destination**, click **Next**.

7 In **Destination Options**, verify that **Import Operation Type** is set to **create/update**.

8 Click **Next**.

9 Click **Execute Migration**.

Run Update Native Directory Utility

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

Product-Specific Updates

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

Planning

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

Financial Management

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

Reporting and Analysis

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

See *EPM_ORACLE_INSTANCE*/bin/ReportingAnalysis/syncCSSId/ReadmeSyncCSSId_BI.txt for detailed instructions to run the syncCSSId utility.

Glossary

access permissions A set of operations that a user can perform on a resource.

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

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

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

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

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

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

backup A duplicate copy of an application instance.

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

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

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

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

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

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

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

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

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

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

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

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

manual stage A stage that requires human intervention.

migration The process ofcopying applications, artifacts, or users from one environment or computer to another; for example, from a testing environment to a production environment.

migration audit report A report generated from the migration log that provides tracking information for an application migration.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

sync Synchronization of Shared Services and application models.

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

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

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

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

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

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

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

token An encrypted identification of one valid user or group on an external authentication system.

transformation 1) A process that transforms artifacts so that they function properly in the destination environment after application migration; 2) In data mining, the modification of data (bidirectionally) flowing between the cells in the cube and the algorithm.

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

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

Index

I

K

L

M

N

O

P

R

S

T

U

W