Oracle Utilities Application Framework

Security Guide

Release 4.5.0.1.3

F88532-01

For use with:

Oracle Utilities Customer Care and Billing v2.9.0.1.3
Oracle Utilities Customer to Meter v2.9.0.1.3
Oracle Utilities Digital Asset Management v2.0.0.1.3

December 2023

ORACLE

Oracle Utilities Application Framework 4.5.0.1.3 Security Guide

Contents

Chapter 7

Chapter 8

Chapter 9

Chapter 10

Chapter 11

Chapter 1

Oracle Utilities Application Framework Security Guide

Welcome to the Oracle Utilities Application Framework Security Guide.

This document describes how you can configure security by using the default features. It includes the following:

This guide applies to the following products:

- Oracle Utilities Customer Care and Billing v2.9.0.1.3
- Oracle Utilities Customer to Meter v2.9.0.1.3
- Oracle Utilities Digital Asset Management v2.0.0.1.3

Note: Oracle Utilities Application Framework refers to the underlying framework on which different Oracle Utilities applications are built. In this document, we use that term when talking about functionalities or features that apply for all the applications listed above.

Related Documents

For more security-related information, see these Oracle resources:

- Oracle Utilities Application Framework Server Administration Guide
- Oracle Utilities Application Framework Advanced Security (Doc Id: 1375615.1)
- Technical Best Practices (Doc Id: 560367.1)
- Batch Best Practices (Doc Id: 836362.1)
- Database Vault Integration (Doc Id: 1290700.1)
- Oracle Identity Management Suite Integration with Oracle Utilities Application Framework based products (Doc Id: 1375600.1)
- Web Services Best Practices (Doc Id: 2214375.1)
- Audit Vault Integration (Doc Id: 1606764.1)
- Oracle Utilities SaaS Cloud Security (Doc Id: 2595978.1)
- These documents are available from My Oracle Support and/or Oracle Documentation.

Critical Patches

Oracle recommends that customers get all their security vulnerability information from Oracle Critical Patch Update Advisories available from links at Critical Patches, Security Alerts and Bulletins. It is strongly recommended that all critical patches be applied in a timely manner.

Additional full details are available at Oracle Software Security Assurance.

Chapter 2

What's New In Security

With each release of the product, new and improved security features are made available. Refer to the release notes provided with your product on the Oracle Energy and Water Documentation page on Oracle Help Center for additional advice.

Chapter 3

Introducing Security

One of the key aspects of Oracle Utilities Application Framework is security, not only to confirm the identity of an individual user, but also to determine data and functions within the application.

This chapter includes the following:

Security Features

Security is one of the key features of Oracle Utilities Application Framework architecture, since it protects the access to your application, its functionality, and the underlying data stored and managed via Oracle Utilities Application Framework.

From an architecture point of view the following summarizes the approach to security:

- **Web Based Authentication:** Oracle Utilities Application Framework provides a default method, using a traditional challenge and response mechanism, to authenticate users.
- **Support for Oracle WebLogic security:** Oracle WebLogic can integrate into several internal and external security stores to provide authentication services. Oracle Utilities Application Framework can use those configurations, to liaise via Oracle WebLogic, to authenticate users for online and Web Services based security.
- **Operating System Security:** For non-online and non-web service-based channels, Oracle Utilities Application Framework utilizes the operating system security (including any additional products used to enhance the base operating system security).
- **Non-Cookie based security:** After authentication the user's credentials form part of each transaction call to correctly identify the user to the internal authorization model to ensure the user is only performing permitted actions. This support is not browser cookie-based.
- **Secure Transport Support:** Transmission of data across the network can utilize the secure encryption methods supported for the infrastructure.
- **Inter-component security:** Calls within Oracle Utilities Application Framework and across the tiers are subject to security controls to ensure only valid authenticated and authorized users using Java Authentication and Authorization Services (JAAS).
- **Inbuilt Authorization Model:** Once a user is authenticated then the internal authorization model is used to determine the functions and data the user has access to within Oracle Utilities Application Framework.
- **Native Web Services Security:** Web Services available from Oracle Utilities Application Framework are natively available from Oracle WebLogic. A wide range of security policies are available.
- **Keystore Support:** Keys for encryption can be externalized in JCEKS based keystore.
- **Integration with other security products:** Implementation of security varies from customer to customer, so Oracle Utilities Application Framework allows integration of other security products to offer enhanced security implementations, either directly or indirectly.

Additional Security Resources

In addition to the security resources described in this guide, Oracle Utilities Application Framework provides the following additional security resources:

- **Oracle Database Vault:** Oracle Database Vault provides fine-grained access control to your sensitive data, including protecting data from privileged users. Oracle Database Vault Administrator's Guide and Database Vault Integration (Doc Id: 1290700.1) describes how to use Oracle Database Vault.
- **Oracle Audit Vault:** Oracle Audit Vault collects database audit data from sources such as Oracle Database audit trail tables, database operating system audit files, and database redo logs. Using Oracle Audit Vault, you can create alerts on suspicious activities, and create reports on the history of privileged user changes, schema modifications, and even data-level access. Oracle Audit Vault Administrator's Guide explains how to administer Oracle Audit Vault and Audit Vault Integration (Doc Id: 1606764.1).
- **Oracle Advanced Security:** See Oracle Database Advanced Security Administrator's Guide for information about advanced features such as transparent data encryption, wallet management, network encryption, and the RADIUS, Kerberos, Secure Sockets Layer authentication.
- **Oracle Identity Management Suite:** Oracle offers a range of specialist security products to manage user identities, password management, single sign on, access management, identity governance, fraud detection and directory services. The Oracle Identity Management Suite Administrator Guides and Oracle Identity Management Suite Integration with Oracle Utilities Application Framework based products (Doc Id: 1375600.1) provides additional information about these products and integration capabilities.

Chapter 4

Authentication

From a security point of view authentication is about identification of the user. It is the first line of defense in any security solution. In simple terms it can be as simple as the challenge-response mechanism we know as userid and password. It can be also as complex as using digital certificates as the identification mechanism and numerous other schemes for user identification.

The authentication aspect of security for Oracle Utilities Application Framework is delegated to the infrastructure used to run the product. This is due to the following:

- **Authentication scheme support:** Oracle WebLogic supports several industry standard security repositories and authentication methods. These can be native to Oracle WebLogic or additional products that can be integrated.
- **Enterprise Level Identity Management:** Identity Management is typically performed at an enterprise level rather than managed at an individual product level. Oracle Utilities Application Framework typically is not the only application used at any site and managing security across the enterprise is more efficient.

The following topics are included:

Online Authentication

Oracle Utilities Application Framework delegates the responsibility of authentication of the online users to Oracle WebLogic. This means that any integration that Oracle WebLogic has with specific security protocols or security products can be used with the product for authentication purposes. The configuration of authentication is therefore performed within Oracle WebLogic itself.

Typically, Oracle WebLogic support one or more of the following:

- **Inbuilt Security:** Oracle WebLogic supplies a default basic security store and associated security management capability that can be used if no other security repository exists.
- **LDAP Based Security:** The Lightweight Directory Access Protocol (LDAP) is a protocol for accessing and maintaining distributed directory information services. LDAP is used to standardize the interface to common security repositories (such as Oracle Internet Directory, Microsoft Active Directory etc). LDAP support may be direct or indirect via Identity Management software like Oracle Virtual Directory or Oracle Identity Federation.
- **SAML Based Security:** Security Assertion Markup Language (SAML) is an XML based data format for exchanging authentication and authorization information between parties.
- **DBMS Based Security:** Oracle WebLogic can store, manage and retrieve security information directly from a database.
- **Operating System Based Security:** Oracle WebLogic can store, manage and retrieve security information directly from the underlying operating system.
- **Oracle Utilities Application Framework Security Provider:** The Oracle Utilities Application Framework includes an optional custom WebLogic Security Provider that allows implementations to verify user identity and whether the user is enabled as part of a security domain configuration. If the security provider is not used, these attributes are checked at login time.

These security configurations can be natively support or can be augmented with additional products. Refer to the Security Guides supplied with Oracle WebLogic for details of the security configuration process.

Batch Authentication

The Batch component of the architecture utilizes the operating system-based security (including any extensions to that security) to authenticate users to execute batch processes. From an authentication point of view:

Batch users must be defined in the operating system and associated with the operating system security group assigned at product installation time. This ensures users have appropriate access to product resources and the ability to write logs.

Threadpools can be started by any valid operating system user but ideally threadpools and submitters should be executed by the same operating system user.

Before any threadpool or submitter is executed, the user must execute the splenviron utility to set the environment variables for the application correctly. This can be done at

the command line for each threadpool and submitter or globally using the logon profile for the operating system user.

Web Service Authentication

The Web Service component of Oracle Utilities Application Framework is housed in Oracle WebLogic and utilizes the native Web Services security mechanism supported by that server. From an authentication point of view:

- The Web Service is deployed using an administration account using the utilities provided from the product online (for developers) or using command line utilities.
- The Web Service is managed using the administration account using the administration console provided with Oracle WebLogic.
- Oracle WebLogic allows security policies and security access rules to be configured at an individual Web Service point of view. Any of the valid policies and security rules supported by Oracle WebLogic can be used.
- Web Service management products such as Oracle Web Services Manager can be used to augment security for Inbound Web Services.

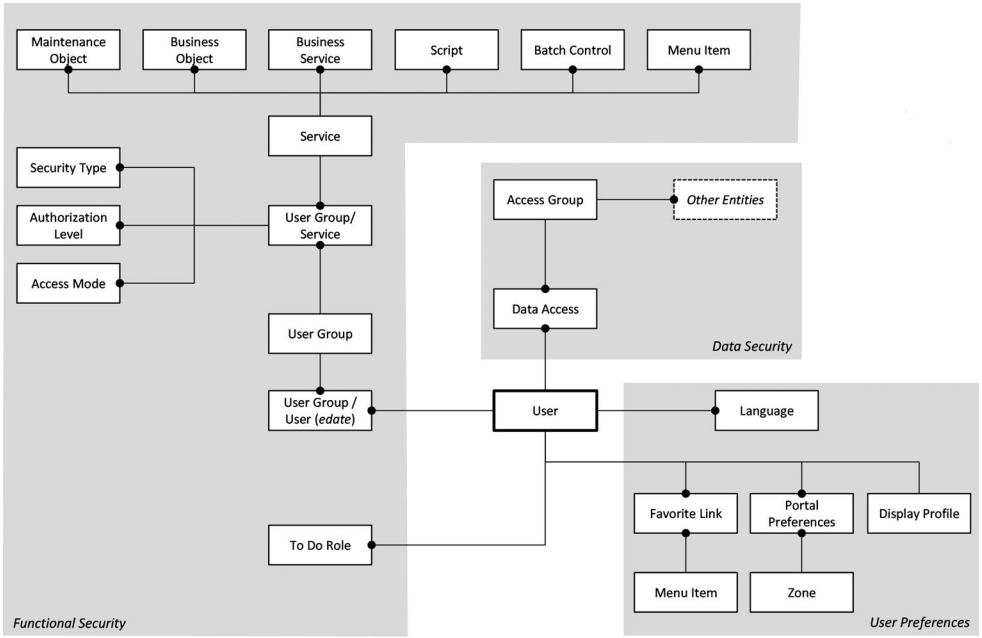
Chapter 5

Authorization

Once a user is identified, they must be authorized to specific functions and data within Oracle Utilities Application Framework, which uses an inbuilt security model for authorization. This model contains all the data necessary for the definition of authorizations to function and data.

Authorization Model

The following data model describes the security authorization model of Oracle Utilities Application Framework.



Users

A record of each user is stored in the User entity, which defines the attributes of the user including identifier, name, Portal preferences, Favorites, Display Profile (such as format of dates and so on), Language used for screens and messages, and other attributes. Users are attached to To Do Roles that allow the user to process any error records for

background processes. For example, if the XXX background process produces an error, it is possible to define the users that will process and address it.

User Groups

User Groups are a mechanism for grouping users, usually around job roles. Users are attached to User Groups through a relationship that is effective-dated, which means that the date period it is active across is also defined. This can be useful for the attachment of temporary employees such as contractors or for people who change roles regularly.

Each User Group is authorized to access certain Application Services, which are the functions within your application. Loosely, these correspond to each of the accessible screens. Application Services have valid Access Modes with standards being Add, Modify, Read and Delete.

Additionally, it is possible to define authorization levels to functions for the User Groups. For example, you may find that a certain group of users can only approve payments of a certain level unless additional authorization is obtained. The Authorization Level is associated with a Security Type, which defines the rules for a given Application Service.

> **Note**: To use Security Types, the implementation must develop server side or client-side user exits to implement code necessary to implement the security level.

Services can be attached to individual Menus, Batch Controls, Maintenance Objects, Business Objects, Business Services and Scripts to denote the service to be used to link user groups to access these objects. In this case Business Object security overrides any Maintenance Object security. The same applies to Business Services security overriding the Application Service it is based upon.

The Oracle Utilities Application Framework allows you to limit a user's access to specific data entities to prevent users without appropriate rights from accessing specific data. By granting a user access rights to an account, you are granting the user access rights to the account's bills, payment, adjustments, orders, and so on.

Access Groups

An Access Group defines a group of accounts that have the same type of security restrictions. A Data Access Role defines a group of Users that have the same access rights (in respect of access to entities that include access roles). When you grant a data access role rights to an access group, you are giving all users in the data access role rights to all entities in the access group.

The following points summarize the data relationships involved with data security:

- Entities reference a single access group. An access group may be linked to an unlimited number of relevant entities.
- A data access role has one or more users associated with it. A user may belong to many data access roles.
- A data access role may be linked to one or more access group. An access group may be linked to one or more data access roles.
- Information in the security model can be manually entered using online transactions and also can be imported and synchronized using a LDAP import

function provided with the Web Services Adapter. The latter is typically used with customers who have lots of online users to manage.

- The authorization model is used by all modes of access to the product. Native interfaces (java classes) are used by all objects and a PL/SQL procedure is provided for reporting interfaces.

Chapter 6

Managing Security

Once the security definitions are established, they must be managed from the application itself, using the security infrastructure and security repositories.

The following topics are included:

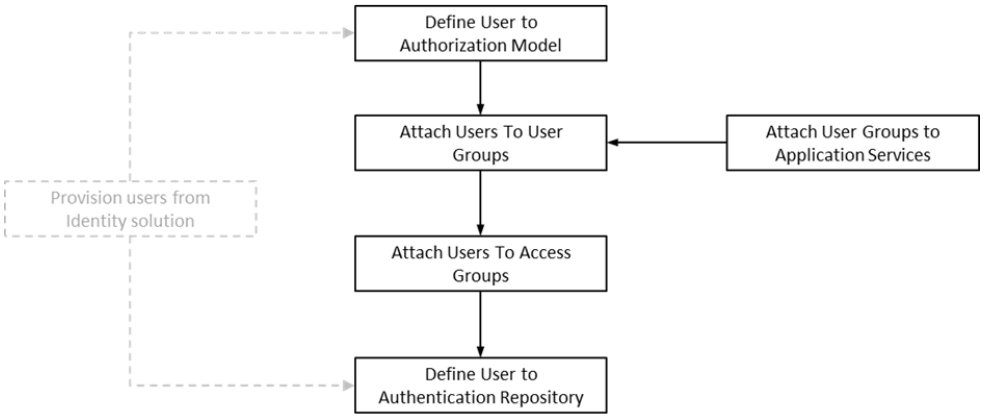
- Online User Management
- Managing Batch Users
- Managing Web Services Users
- User Authentication

Online User Management

To manage online users, several facilities must be configured:

- The security repository and rules must be configured in Oracle WebLogic to enable authentication. Refer to Oracle WebLogic Administration Guides for more information.
- The product group used to connect users to Oracle WebLogic resources should be created in the security repository and configured in the product configuration. The default value for this setting is cisusers. Refer to the Server Administration Guide for more information on this setting.
- Users need to be connected to the product group within the security repository to indicate that they can access the Oracle WebLogic resources.

The process for managing online users is outlined in the following process:



- Users should be defined to the authorization model to define their profile and permissions within the application. Refer to the User Management section of this chapter for more details.
- Attach user groups to Application Services to define the subset of service and actions valid for that group of users. Refer to Defining User Groups to Application Services for more details of this process.
- Attach Data Access Groups to the users. This defines the subset of data that the user has access to. Refer to Defining Users to Data Access Groups for more details of this process.
- Attach users to the appropriate user groups to define the subset services and valid actions the user can perform within your application. Refer to Defining Users to User Groups for more details of this process.

User Management

The User object in your application is used to record the security information used for identification of the user and their permissions.

Oracle Utilities Application Framework provides a maintenance function to maintain these definitions within the product. To maintain the users the following is performed:

- Navigate to the user menu option on the Administration menu. Using the Add User option on the menu allows navigation to the add function.

- The User maintenance object is displayed which maintains the security information for a user.

Field	Comments
Userid	This is the unique user identifier used within the product used for authorization activities. Limited to eight (8) characters in length.
Login Id	This is the unique user identifier used within the product used for authentication purposes. This must match the value used in the security repository to successfully use the product. Limited to 256 characters in length. This value can be the same or different to the Userid.
Last Name	Last Name of user. Limited to 50 characters in length.
First Name	First Name of user. Limited to 50 characters in length.
User Enable	Whether the user is active in the security system or not. Valid Values: Yes (default) – User is active and can use the system, No – User is disabled and cannot use the system. Refer to User Enable and Disable for more details.
User Type	The type of user. Valid Values: Blank = Normal user, Template = Template Users.
Language	Default Language used for user. For non-English languages, Language pack must be installed to use specific languages.
Display Profile Id	The display profile associated with the user. This controls the display of currency, dates, and so on.
Time Zone	Time Zone allocated to user account. **Note**: This feature is only applicable to specific products. Check your product online documentation for more details about applicability.
Email Address	Optional Email address associated with user. This is used by utilities and can be used for interfaces requiring email addresses.
Dashboard Width	Default width for Dashboard Portal. Setting this value to zero (0) will disable the dashboard altogether.
Dashboard Location	Preferred location of the Dashboard Portal. This capability is only enabled for relevant platforms only.
Dashboard State	Preferred initial state of the Dashboard Portal. This capability is only enabled for relevant platforms only.
Home Page	The default home page associated with the user.
Portals Profile User Id	The userid used to inherit portal definitions from. Refer to Template Users for more information.
Favorites Profile User Id	The userid used to inherit favorite definitions from. Refer to Template Users for more information.
To Do Summary Age Bar	The settings for the color coding of the To Do Summary portal in the dashboard. This can be used to indicate relative age of to do entries.

Field	Comments
User Groups	This is a list of user groups and their associated expiration dates. Refer to Defining Users to User Groups for more information.

- Save the additions/changes for the user using the Save function on the top of the screen.

Template Users

By default, portal preferences and favorites are set at an individual user level. It is possible to inherit the portal and favorites from other users to reduce the maintenance effort for security information. Changes to the profile user are automatically inherited to any users where the profile user is attached to.

To use this functionality the following must be performed:

- Set up each user to be used as a template and set the **User Type** to *Template*.
- For any user that will inherit the portal preferences and favorites, specify the appropriate template user in the following fields:
  - **Portal Preferences:** Use the Portals Profile User Id to specify the Template User to be used to inherit the portal preferences.
  - **Favorites:** Use the Favorites Profile User Id to specify the Template User to be used to inherit the favorites preferences.

Once any changes are made to the Template Users' portal preferences and favorites, these changes automatically apply to any attached users.

Assigning To Do Types

Note: To Do records can be assigned to explicit users or groups of users. This section covers the latter condition.

Note: Refer to the online Administration Help for a discussion about the To Do function. To Do Roles must be set up prior to using this function.

Your application generates To Do records for any function or error condition that requires human intervention. The To Do record contains a type and role to be used assist in assigning the appropriate resources to work on the condition indicated by the To Do.

For security purposes, users need to be attached to the relevant roles for the To Do facility to limit which To Do Types an individual user can work upon. To manage the To Do Roles to be assigned to users, navigate to the **To Do Roles** tab of the User Maintenance function and select the **Add** or **Delete** icon. You can use the **Search** icon to find existing To Do Roles. Once users have been attached to the To Do Roles, they can access the associated To Do Types assigned to the role or any To Do directly assigned to them.

Assigning User Portal Preferences

Note: Refer to the online Administration Help for a discussion about the Portal/Zone functionality. Portals and Zones must be setup prior to using this functionality.

Note: Portal Preferences can be inherited from other users if Template Users are used. In this case the ability to set for portal preferences for users attached to a template user are disabled.

The user interface of your application is made up of Portals containing individual Zones. Each of the portals and zones can be associated with an application service for security purposes. Users that are attached to User Groups that are also attached to those Application Services can view and use the portals and zones.

The order of display and other factors are defined at an individual user basis. To define the portal preferences for a user, navigate to the Portal Preferences tab of User Maintenance function. This will display a screen with a list of the portals the user has access to, via the user groups they are attached to.

To maintain the preferences for a specific portal, expand the portal entry in the list by clicking the name of the portal or using the Expand All functionality. For example:

The following zone preferences can be set for the user:

Zone Preference	Description
Display	Whether the zone is included or not in the portal. This allows specific zones to be displayed at startup time while other zones can be hidden and only displayed upon conditions in other zones. See Zone Visibility in the online Administration guide for more information.
Initially Collapsed	Whether the zone is displayed collapsed on initial load. Zones are only executed when they are expanded. Marking zones as Initially Collapsed can prevent them from being executed and can speed up portal rendering times.
Sequence	Defines the relative order of the zones within the portal. A value of zero (0) takes the default sequence from the portal definition.
Refresh Seconds	Defines the zone auto refresh rate (this is only applicable to a subset of zone types). A value of zero (0) disables auto-refresh.
Security Access	This is an information field that indicates whether the user has access to the zone or not (while unlikely, it is possible to have a portal contain zones not permitted for access to an individual user). Refer to the online documentation for more information.

Assigning Bookmarks

You can attach bookmarks to your user profile to access pages including the context of the pages. You can use the **Bookmark** button to define bookmarks that attach the page and context to the user profile.

Note: Bookmarks are added at runtime by end users using the **Bookmark** button. This function only displays or deletes the bookmarks assigned by the user.

It is possible to view and remove bookmarks on the user profile by navigating to the Bookmarks tab of the User Maintenance function. You can set your bookmark preferences through the following fields:

Field	Description
Sequence	Internal sequence used for sorting.
Name	The name of the bookmark. The URL for the bookmark is hidden and not editable.

You can use the **Delete** icon to remove existing bookmarks from your list.

Assigning Favorite Links

Users can set several favorite functions or menu items that they can access using keyboard shortcuts or via the **Favorites** zone on the Dashboard.

> **Note**: Favorites can be inherited from other users if Template Users are used.

Configuration of favorite functions or menu items is through the **Favorite Links** tab of the User Maintenance function. Users can set favorite link preferences through the following fields:

Field	Description
Sequence	The relative sequence number of the favorite link used for sorting purposes.
Navigation Option	The Navigation Option to display the favorite links. This can reference the zone or maintenance function to display after selecting the favorite link.
Security Access	Indicates whether the Navigation Option is accessible or not to the user.

To manage the Favorites to be assigned to users, select the **Add** icon to assign the favorite link with the appropriate Navigation Option and Sequence or select the **Delete** icon to remove an existing Navigation Option from the list. You can use the **Search** icon to find existing Navigation Options.

Favorites are then available to be displayed in the Favorites zone on the Dashboard.

Assigning Favorite Scripts

Users can set several Favorite BPA Scripts that they can access using the **Favorite Scripts** zone on the Dashboard.

> **Note**: Favorites can be inherited from other users if Template Users are used.

Configuration of favorite scripts is through the **Favorite Scripts** tab of the User Maintenance function. Users can set favorite script preferences through the following fields:

Field	Description
Sequence	The relative sequence number of the favorite used for sorting purposes.

Field	Description
Script	The BPA Script to use in order to display the favorite function or menu items.
Security Access	Indicates whether the BPA Script is accessible or not to the user.

To manage the Favorites to be assigned to users, select the **Add** icon to assign the favorite link with the appropriate Script and Sequence or select the **Delete** icon to remove an existing Script from the list. You can use the **Search** icon to find existing BPA scripts.

Favorites are then available to be displayed in the Favorite Scripts zone on the Dashboard.

Assigning User Characteristics

Your application can extend objects within itself through Characteristics, which act as additional data attributes for providing supplementary information or custom algorithms for processing.

> **Note**: Oracle Utilities Application Framework applications ship with a predefined set of Characteristic Types.To use User Characteristics, the appropriate characteristic types must be created and attached to the User object. Refer to the online Administration documentation for more information.

The User object in your application can also be customized using Characteristics. This can be achieved by navigating to the **Characteristics** tab of the User Maintenance function. The following fields can be set for the favorites:

Field	Description
Characteristic Type	The characteristic type associated with the User object. This is a drop-down list of the valid characteristic types associated with the object.
Sequence	The relative sequence number of the characteristic used for processing purposes.
Characteristic Value	Depending on the configuration of the Characteristic Type, this value may be free-formatted, an attachment, in a specific format or a specific set of values.

To manage Characteristics to be assigned to users, select the **Add** icon to assign the characteristic (indicating Characteristic Type) with the appropriate Sequence or select the **Delete** icon to remove an existing characteristic from the list.

Defining Users to User Groups

Access to the application services requires User Group connections that are connected to Application Services. The connections define the linkage for functions that are accessible to users.

The attributes of the user to user groups links are as follows:

- The link is subject to an expiration date to allow representation of transient security configurations.
- Each link is owned and subject to Data Ownership Rules. By default, all site-created links are owned as Customer Modifications.
- User Groups are set up according to site preferences. These can be job-related, organization level-related or a combination of factors.
- Users must be members of User Groups to access the system. A user can be a member of multiple groups.
- Users can be members of groups with overlapping permissions to Application Services. In cases of overlapping permissions, the highest valid permission is used.

You can manage the user and User Group link by navigating to the **Main** tab of the User Maintenance function. You can use the **Add** icon to insert a User Group with the appropriate expiration date or use the **Delete** icon to remove existing User Groups from the list. Use the **Calendar** icon to select the expiration date and set the link's effective date. Use the **Context Menu** icon to navigate to the User Group details to review more information. The user's security is referenced for menu and function access regardless of the access channel (online, web service, or batch) used.

Defining User Groups to Application Services

One of this application's fundamental security configurations is to define User Groups to Application Services. An Application Service can represent an Oracle Utilities Application Framework service, a menu or an object. Linking a User Group to a service allows Access Mode configuration, which defines the valid actions the User Group can perform.

> **Note**: Oracle Utilities Application Framework applications ship with all the Application Services predefined for base functions. These can be used or replaced with custom definitions. A starter set of User Groups that can be used as basis for further security User Groups is loaded with your application.

Additionally, each service can specify Security Types that allow for custom security rules to be applied at runtime. Refer to Security Types for supplementary information.

There are two methods to maintain the links established between User Groups and Application Services: the Application Service Portal and the User Group Maintenance. These are valid for most sites and can be used to manage information distinctively.

Application Service Portal

The Application Service Portal enables you to define an Application Service and to set its Access Modes and User Groups.

You can configure the following **Main** tab settings by navigating to the **Application Service** option on the **Administration** menu:

You can also configure the following zones in the **Application Security** tab to display the User Group membership and manage that relationships:

After granting access to User Groups, you can set the Access Mode and security group specification for the User Group:

Setting	Description
Application Service	The unique identifies of the Application Service used in configuration of security on objects, menus, services, and so on. For custom definitions, Oracle recommends adding a "CM" prefix to distinguish these from Application Services provided by Oracle Utilities Application Framework.
Description	A brief description for documentation purposes that appears on security screens when the Application Service is specified.
Access Modes	Lists the valid modes to access the Application Service. The modes must match the internal actions supported by the objects used by the Application Service. Use the **Add** icon to insert an Access Mode. Note that an Access Mode can only be defined once on an Application Service. Use the **Delete** icon to remove an existing Access Mode from the list. The Access Mode link to the Application Service is ownership-controlled and by default all created links are owned as Customer Modifications. Refer to Data Ownership Rules for more information.

Zone	Description
Application Service Details	Summarizes the Access Modes and Security Types of the Application Service.
User Groups With Access	Lists the User Groups with access to the Application Service, along with the associated expiration dates, Access Modes, Security Types and associated authorization levels. Use the Deny Access function to limit the access of User Groups to the Application Service.
User Groups Without Access	Lists those User Groups without access to the Application Service. Use the Grant Access function to allow User Groups to access the Application Service.

Field	Description
Expiration Date	Specifies the date when access to the User Group expires.
Access Mode	Shows the Access Mode as defined on the Application Service definition. Use the Add icon to insert an Access Mode or use the Delete icon to remove an existing Access Mode from the list.
Owner	Ownership of link. Refer to Data Ownership Rules for more information.
Security Type	The Security Type code associated with the Application Service. Use the **Add** icon to insert a Security Type or use the **Delete** icon to remove an existing one from the list.
Authorization Level	The Authorization Level assigned to the User Group when running the Application Service for the Security Type.

User Group Maintenance

The User Group Maintenance allows you to define the Application Services that User Groups can access and to connect users to User Groups. You can manage the User

Groups by navigating to **Administration**, selecting the **User Group** menu item, and performing the following actions:

- Use the **Context Menu** icon to edit existing permissions.
- Use the **Delete** icon to remove the association between User Group and Application Service.
- Use the **Add** icon to associate a User Group with an Application Service.

Adding or editing associations automatically displays the **Application Services** tab, which enables you to maintain the Access Modes and Security Types for the association through the following fields:

Field	Description
Expiration Date	Indicates the date on which access to the User Group expires.
Access Mode	Valid Access mode as defined on Application Service definition. Use the **Add** icon to add a new Access Mode or use the **Delete** icon to remove an existing Access Mode from the list.
Owner	Ownership of the link. Refer to Data Ownership Rules for more information.
Security Type	The Security Type code associated with the Application Service. Use the **Add** icon to insert a Security Type or use the **Delete** icon to remove an existing one from the list.
Authorization Level	The Authorization Level assigned to the User Group when running this Application Service for the Security Type.

You can also manage users associated with the User Groups through the Users tab fields:

Field	Description
User	The authorization user identifier to associate with the user group.
Expiration Date	Indicates the date on which the association between the user and the User Group expires.
Owner	Ownership of the link. Refer to Data Ownership Rules for more information.

Defining Users to Data Access Groups

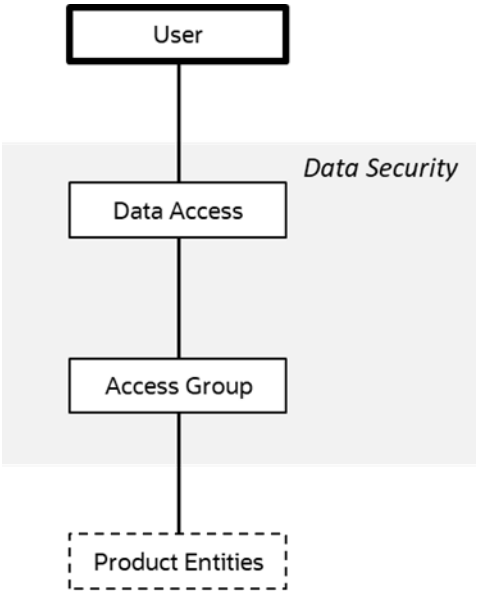
Data Access Groups define the subset of data objects accessible to users. The levels of data access definition are as follows:

- **Data Access Roles:** User are connected to Data Access Roles which defines the groups of data permissions the user has access to. Data Access Roles are connected to Data Access Groups (a.k.a. Access Groups).
- **Data Access Groups:** Data Access Groups are tags that are attached to entities in your application to implement data security. Data Access Groups are maintained using Access Group maintenance. Refer to the online Administration Guide for more details of this facility.

Note: Only some services support Data Access Roles and Data Access Groups. Refer to the online Administration Guide for more details.

Note: Attaching a Data Access Group to a product entity does not automatically implement data security. Queries for that object must be altered to consider the Data Access Group. Refer to the Oracle Utilities SDK for more details.

The figure below illustrates the relationship between Data Access Roles and Data Access Groups:



You can maintain Data Access Roles and Data Access Groups in the **Access Security** tab of the User Maintenance function. Use the **Add** icon to insert a Data Access Role and configure its settings or use the **Delete** icon to remove an existing Data Access Role from the list. The screen will allow the definition and display of the following information:

Field	Description
Default Access Group	When this user creates a new object that is subject to Access Security then this default is used for the value of the Access Group of the new object. This can be overridden by logic within the object if necessary.
Data Access Role	List of Data Access Roles this user is attached to.
Expiration Date	Indicates the date on which the association between the user and the Data Access Role expires.

User Enable and Disable

One feature of security is to attach user records to some objects (automatic or configurable) for audit purposes. You cannot delete a user record if the user performs any work within your application, or is attached to some audit objects across Oracle Utilities Application Framework.

The User Enable function on the User object allows you to activate or deactivate a user by setting the appropriate value for **User Enable**, which has the following implications:

Value	Implications
Enable	• The user can access the system. • The user can process records according to the authorization model. • The user must be active in the Security Repository to fully access the application.
Disable	• The user cannot access the system regardless of the security setup. • The user record is retained for audit purposes only. • The user does not have to exist in the Security Repository. • The key use cases for this option are as follows: • **Support for personnel (permanent or temporary) leaving:** Manually deactivate users once they leave the organization yet keep the information for auditing purposes. • **Logical deletion:** If the user needs to be deleted for any reason, selecting this option removes the user record, preventing access to the system. • **Temporary disablement:** If business rules need to isolate the user record, selecting this option for the appropriate users can effectively deactivate their access to the application. **Note**: Deactivation of the user record will take effect when the user logs in to the system or after the security cache refreshes.

Advanced User Management

The **User Group Portal** supports multiple actions including:

- Setting an expiration date across multiple user group access modes for multiple application services.
- Removing multiple access modes for multiple application services from user groups.
- Adding multiple permissions from multiple application services.
- Maintaining multiple security types across multiple application services

Managing Batch Users

Each time a batch process is executed, the security components of your application must authenticate the user against a security repository and authorize the user to access the components the batch process needs to complete its operations.

The Batch component of the architecture uses the following security mechanisms:

Security Mechanism	Description
Authentication	Any batch users must be defined to the operating system configured security repository and be a member of the operating system group assigned to your application. **Note:** The Userid does not have to match the authorization user.

Security Mechanism	Description
Authorization	The authorization user is defined within the application as per the Online Users and is specified as a job parameter at execution time or in configuration files supplied for the batch process. Refer to the *Server Administration Guide* for details of the parameters used for batch processing.

To manage Batch Users therefore the following is recommended:

- Add the authentication user used to initiate the thread pool and submitter processes for a batch process to the configured operating system repository.
- Specify a valid user authorization identifier as a parameter for the batch process. This identifier must be authorized to the valid actions against the main objects used in the batch process. Refer to the product functional documentation on the objects used in each of the product batch processes.

Managing Web Services Users

From a product perspective, a Web Service is a channel into the objects within your application. Any of the objects, services and scripts available in the product can be exposed as JAX-WS 2.0 based Web Service. From a security perspective Web Services uses the following security mechanisms:

Security Mechanism	Description
Authentication	The Web Services component of Oracle Utilities Application Framework uses the Web Services support native to Oracle WebLogic. This allows security tokens supporting many standards to be used to authenticate individual web service calls.
Authorization	The Web Services component uses the same authorization model as the Online Users and the Batch component. **Note:** The Authorization User within the User object is mapped through the Authentication User in the same way that Online Users are mapped.

To manage Web Services security users the following is recommended:

- Users for authentication are added to the security repository configured with Oracle WebLogic. This should match the Login Id used for the authorization model.
- Security Policies need to be attached to Web Services using Oracle WebLogic. For Oracle WebLogic the security policies available using Oracle Web Services Manager are available for use with individual Web Services. Multiple policies are supported. Refer to the Securing WebLogic Web Services for Oracle WebLogic Server for more information and the policies available.
- Users must be defined to the authorization model with appropriate access to underlying services used by the Web Service. For Web Services based upon business objects, services and scripts, users need appropriate access to the Application Service defined on these objects.

- Transaction Types in the Web Services translate to Access Modes within the Application Service calls.

For more information about Inbound Web Services, refer to the Web Services Best Practices (Doc Id: 2214375.1).

User Authentication

There are two different user identifiers with distinct roles in the User object: Userid and Login Id.

Identifier	Characteristics
Userid	• Used internally for authorization. • Passed to the database connection as the `CLIENT_IDENTIFIER` on the database connection. • Used for record ownership in some objects and in auditing, whereby it cannot be changed after the creation of any records by the user. • The maximum length of Userid is 8 characters.
Login Id	• Used for authentication to the security repository configured on Oracle WebLogic domain. • It can match the Userid or differ to reflect site standards. Unlike the Userid, the Login Id can be changed at any time to reflect changes in the organization such as name changes or acquisition. • The maximum length of Login Id is 256 characters. **Note:** The Login Id must match, in the same case, as the entry in the configured security repository for Oracle WebLogic.

When maintaining a user, it is important that the Login Id is only changed using the maintenance function, LDAP Import or any Inbound Web Service based upon the User object and not directly using other means (such as direct SQL). This due to the fact that a security hash is generated at maintenance time and is checked at login time. At application login time, if the security hash does not match, the user is not authorized to access your application. To ensure security hashes are correct, use the Synchronize Data Encryption function to reset the user security hash.

Chapter 7

Advanced Security

While the default security settings are adequate for most sites, there are several additional advanced settings that can be configured to support a wider range of security requirements.

This section outlines the various security settings available and the configurations supported, including:

- Domain Authentication Group
- Logon Configuration
- Data Ownership Rules
- Configuring JMX Security
- Menu Security Guidelines
- Security Types
- Default Generic Application Services
- Administration Delegation
- Secure Communications (SSL)
- Data Masking Support
- Securing Files
- Password Management
- Securing Online Debug Mode
- Securing Online Cache Management
- Web Services Security
- Message Driven Bean Security
- SOAP Security
- Groovy Support
- Oracle Cloud Object Storage Support
- HTTP Proxy Support
- SYSUSER Account

Domain Authentication Group

The default installation of your application includes a default Authentication Group (`role-name`) defined within the Web Application descriptor (web.xml). This role name is used by the Web Application to link the authorized users within the application to the associated domain physical resources (pages and configuration files) within Oracle WebLogic. The specification of the group in the web descriptor is in the security section.

> **Note**: The security role is used in several sections of the Web Application descriptor.

For example:

```
<security-role>
   <description>OUAF Users</description>
   <role-name>cisusers</role-name>
</security-role>
```

By default, this group is set to `cisusers`, which is configurable for each web component. When the product is deployed to Oracle WebLogic, this group is instantiated ready to be allocated to individual users. Users of the product must be attached to this group to use it.

From a configuration point of view there are several options for this setting:

- The default group may be changed at installation and configuration time using the configuration settings as shown below as outlined in the *Server Administration Guide*. The group name should have no embedded blanks.

Component	Principal Name	Role Name
Online/Help	`WEB_PRINCIPAL_NAME`	`WEB_ROLE_NAME`
AppViewer	`WEB_APPVIEWER_PRINCIPAL_NAME`	`WEB_APPVIEWER_ROLE_NAME`

- If Oracle WebLogic is configured to use an external security repository the configured administration group must exist in the security repository and the users must be connected to this group.

  Note: If the domain administration group is changed after installation time, users will need to be migrated to the new domain administration group either manually, using tools provided with the security repository or through Oracle WebLogic.

Logon Configuration

The default configuration for Online Authentication is using a logon screen for the online product, online help and online AppViewer applications. The product supplies a prebuilt logon screen for all three components preconfigured.

At logon it detects that a user has not logged on before (the presence of a `JSESSIONID` cryptographically-secure session cookie issued by the Web Application Server is used). Depending on the configuration (in the web.xml) of the applications, housed in Oracle WebLogic, the following is performed:

- **FORM** - This is the default setting to support a logon screen with an associated error screen in case of unsuccessful logon. Your application provides a prebuilt logon screen but can be replaced with custom logon screens by setting the following configuration settings appropriately for each web component as outlined in the *Server Administration Guide*:

Component	Login Screen	Login Error Screen
Online	WEB_FORM_LOGIN_PAGE	WEB_FORM_LOGIN_ERROR_PAGE
Help	WEB_HELP_FORM_LOGIN_PAGE	WEB_HELP_FORM_LOGIN_ERROR_PAGE
AppViewer	WEB_APPVIEWER_FORM_LOGIN_PAGE	WEB_APPVIEWER_FORM_LOGIN_ERROR_PAGE

Note: Custom logon screens should be placed in the cm directory of the Web Application Server as outlined in the Oracle Utilities SDK.

- **BASIC** - The browser will issue a call to the operating system to display the default logon dialog supplied with the operating system. No logon dialog is supplied.

  Note: BASIC authentication is considered a relatively weak authentication scheme, and therefore is not recommended for use.

- **CLIENT-CERT** - This is an advanced configuration to allow for certificated (one way or two way) to be used. Refer to the Administering Security for Oracle WebLogic Server documentation for more details on the additional configuration required.

Data Ownership Rules

On each of the objects (and on selected child objects) an owner flag is included to determine the origin of the data. The owner flag is used by the application to determine the maintenance owner of key data as well as to protect from accidental deletion important data shipped with the product.

The value of the flag is displayed on maintenance screens to visually indicate the data owner. The location of the information varies from the top left of maintenance pages, within lists of information (to apply to individual rows) and within sections of maintenance pages.

The data ownership flag has the following values:

Value	Description
Base	This is crucial information shipped with your application and cannot be deleted or modified through the **Delete** or **Medication** functions and regardless of user permissions. This value is reserved for use by the application to ship and protect key information. **Note:** To delete this information directly from a product database will cause unexpected results.

Value	Description
Product Name	The name of the product that owns the data. This value is similar to the Base value but indicates which component the data is applicable to. All the rules that apply with the Base value apply to this value.
Customer Modification	This indicates that the data was added by the implementation using the various methods and that it is owned by the implementation. **Note:** Deletion of data is permitted using the valid deletion functions for authorized users.

Note: In general, sites can only maintain Customer Modification owned records. Other ownership values are reserved to protect product installation supplied data.

Configuring JMX Security

The operations interface is based upon Java Management Extensions (JMX), which allows components of the application to be managed and monitored from `JSR160` compliant consoles including `jconsole` or Oracle Enterprise Manager.

Refer to the *Server Administration Guide* and to the *Batch Server Administration Guide* for more details of the JMX operations interface.

By default, the JMX implementation and configuration uses the default simple file based security as outlined in the JMX Specification.

Default Simple File-Based Security

The default configuration is based upon a properties file containing name/value pairs corresponding to role/password pairs and authorization can be also based on a properties file containing name/value pairs corresponding to role/access pairs where access can be any of `readonly` access which grants read access to any remote operation and `readwrite access` which grants access to read and update operations in the interface.

Note: By default, the user `(BSN_JMX_SYSUSER)` and password `(BSN_JMX_SYSPASS)` for the administrator are automatically added to the configuration files.

To use this facility the following file should be maintained using an appropriate editor located in `$SPLBASE/scripts` directory:

- `ouaf.jmx.access.file` – This file contains the Userid and access Permissions in the format separated by a blank space:

Field	Comments
Userid	Authentication user to access JMX.
Permission	Permission assigned to user. Valid values are: `readonly` – No update access. `readwrite` – Update and update operations access.

- `ouaf.jmx.password.file` - This file contains Userid and Password in the format separated by a blank space:

Field	Comments
Userid	Authentication user to access JMX.
Password	Password in plain text or encrypted.

Note: These files are also tailored using custom templates. `ouaf.jmx.access.file.template` and `ouaf.jmx.password.file.template` are used for the configuration.

SSL-Based Security

To secure communications for JMX using the Java SSL support the following process must be performed:

- Security has to be setup using the Default Simple File-Based Security or Other Security Sources.
- A key pair and certificate need to be setup on your server. Refer to the Monitoring and Management Using JMX Technology or to the Oracle WebLogic Administration documentation for details and utilities available for this process.
- Set additional java parameters using the `WEB_ADDITIONAL_OPT` for the online/ Web Services and `BATCH_MEMORY_ADDITIONAL_OPT` for Batch. Refer to the *Server Administration Guide* and *Batch Server Administration Guide* for details of these parameters. The following additional system properties must be set:

System Property	Comments
`javax.net.ssl.keyStore`	Keystore location
`javax.net.ssl.keyStoreType`	Default keystore type
`javax.net.ssl.keyStorePassword`	Default keystore password
`javax.net.ssl.trustStore`	Truststore location
`javax.net.ssl.trustStoreType`	Default truststore type
`javax.net.ssl.trustStorePassword`	Default truststore password
`com.sun.management.jmxremote.ssl`	Set to true
`com.sun.management.jmxremote.registry.ssl`	Set to true
`com.sun.management.jmxremote.ssl.need.client.auth`	Set to true

Note: For a full description of additional options and SSL setup refer to Monitoring and Management Using JMX Technology.

Note: Specification of system properties for java are as per the java command-line.

Note: For sites using Oracle WebLogic in native mode, configuration of SSL requires Configuring SSL in WebLogic Server and altering the startup scripts for Oracle WebLogic to include the above options.

Note: In line with industry standards either HTTP or HTTPS can be used. They cannot be used simultaneously.

Other Security Sources

Whilst, by default, the file-based repository is supported, it is possible to configure the authentication of JMX to use an alternative data source such as an LDAP Server. This involves changing the Java Authentication and Authorization Service (JAAS) configuration stored in the `java.login.config file $SPLEBASE/splapp/config` directory.

In the JAAS configuration file there is a default `jmxrealm` that contains the default JMX `LoginModule`. This can be changed, using custom templates, to support an alternative source for authentication. Refer to the `LdapLoginModule` documentation for information and examples of login configurations.

Note: To implement the custom security source custom templates for `java.login.config` must be implemented according to the process outlined in the *Server Administration Guide*.

Menu Security Guidelines

By default, a menu option is displayed whenever a user has access to the underlying application service definition attached to objects that are indirectly linked to a menu entry. Whilst this behavior is enough for most needs, it is possible to place an override on an individual menu item to override the lower level security levels. This is particularly useful where implementations wish to replace base supplied menu items with custom menu items.

By linking a menu item to a new service that can reference the underlying objects and specifying an Application Service (optionally also including an Access Mode) would override the permissions on the underlying objects.

It is possible to specify the Application Service on a menu item on the Menu Items tab on the Menu option on the Administration menu.

Security Types

By default, users have full access to the objects via the access methods specified in their User Groups. If the implementation is to implement additional levels or rules, then the application service must use Service Types. The definition of a Service Type allows additional tags to be attached to service definitions and then code written to detect and take advantage of the presence of the tag to limit security access to specific object data. For example, whether data is masked or not or some limit is placed on values of data.

To define Security Types, use the **Security Types** menu option on the **Administration** menu to display the Security Types maintenance function.

On this function define the following in relation to the Security Type:

Field	Description
Security Type	Identifier for Security Code.
Description	A brief description of the use of the Security Code.
Authorization Levels	A list of codes (Authorization Level) and associated descriptions. Use the **Add** icon to add a new Authorization Level or use the **Delete** icon to remove an existing Authorization Level from the list. The Authorization Level values are free format but should be representative of the desired function. The Description is used to explain the value.
Application Service Id	A list of associated Application Services to use this Security Code. Use the **Add** icon to add a new Application Service or use the **Delete** icon to remove an existing Application Service from the list.

Note: To fully implement the rules associated with Security Types, code must be included in objects to implement security logic.

Default Generic Application Services

By default, a whole set of Application Services are defined against base functions. In line with Data Ownership Rules, some of these records can be altered and new functions added. A set of generic Application Services are also shipped with your application to provide a mechanism for defining new zones, new objects or new menu items for rapid deployment.

There are two generic Application Services that can be used to secure objects, zones and menu items:

Application Service	Description
F1-DFLTAPS	This is a generic execution Application Service which is designed to secure zones and menu options. It only supports the Execute Access Method.
F1-DFLTS	This is a generic maintenance Application Service which is designed to secure business objects. It supports the Add, Modify, Delete and Inquire Access Methods.

Note: The use of these generic Application Services is optional.

Administration Delegation

By default, your application provides a single administration account, as configured in the SPLADMIN configuration setting, in the ENVIRON.INI configuration file, to manage the operational aspects of the product. This operating system user is the owner of the product when it is installed and is typically used for all operational aspects of the product.

Note: It is not possible to change the product administration account after installation. If this is desired, it is recommended to remove the product and reinstall using the alternative administration account.

Whilst the single administration account is enough for most needs it is possible to provide additional administration accounts to delegate administration tasks. To delegate administration the following must be configured:

- Any administration user must be a member of the operating system group allocated to the product as outlined in the SPLADMINGROUP configuration setting in the ENVIRON.INI configuration file.
- If you are using Oracle WebLogic in native mode, then the console will execute the native facilities to start and stop the product. It is recommended that the user allocated to Oracle WebLogic at installation time be a member of the operating system group outlined in SPLADMINGROUP configuration setting in the ENVIRON.INI configuration file.

  Note: Customers using Oracle Enterprise Manager, with or without Application Management packs, should use the administration delegation and credential management capabilities of that product to manage administration delegations.

Secure Communications (SSL)

By default, your application uses HTTPS to communicate to the browser and across the tiers. The transport protocol can be encrypted using SSL/TLS to secure transmission of data across networks.

> **Note**: Oracle strongly recommends that customers use SSL to secure transmission for production environments.

To implement SSL the following process must be completed:

1. Configure Oracle WebLogic to use the SSL protocol. For Oracle WebLogic customers refer to Configuring SSL in Oracle Fusion Middleware Securing Oracle WebLogic Server.
2. Set the SSL Port Number using the WEB_WLSSLPORT configuration parameter as outlined in the product *Server Administration Guide*.
3. Once the setup has been tested and verified refer to the console documentation on disabling insecure protocols.

Data Masking Support

If data within the object is considered a candidate for data masking, then the masking capabilities with the product can be used to mask the data in an appropriate fashion.

> **Note**: The data is not stored in a masked fashion; it is configured to be displayed in masked format for users using Security Types.

To mask data using the internal data masking capability:

- An internal algorithm type of F1-MASK is supplied with the product to perform basic data masking.
- The following parameters are applicable to the algorithm:

Parameter	Description
Masking Character	The character to be used as a mask. By default, the asterisk (*) character is used.
Number of Unmasked Characters	The number of suffix characters to unmask. Commonly, the last *x* characters are displayed unmasked to allow some identification. A value of zero masks all characters.
Unmasked Characters	List of characters without spaces to leave unmasked. Commonly, this is used to denote delimiter characters to enhance recognition.
Application Service	Used for security authorization checking. It allows global or local services to be configured to indicate security access to data masks.
Security Type	Used to flag which users will view the data in masked on unmasked format. User Groups need to be connected to the Application Service and Security Type and given the Authorization Level to determine the level of data masking.
Authorization Level	The authorization level used that determines if the user can access the unmasked data. All other authorization levels in the Security Type indicate masked data.

- Configure an Algorithm entry of Algorithm Type `F1-MASK` for the desired masking configuration. Algorithm entries can be shared across fields to be masked using the **Algorithm** menu option on the **Administration** menu.
- Attach User Groups to the Application Service with the appropriate Authorization Level for the Security Type.
- Create or update a feature configuration with a Data Masking feature type by using the **Feature Configuration** menu option on the **Administration** menu.
- For each field to mask, add an entry to the **Options** section of **Feature Configuration** and configure the following settings:
  - **Option Type:** Select Field Masking for Data Masking.
  - **Sequence:** Specify a sequence number for sorting purposes.
  - **Value:** Specify a tag string delimited by a comma to indicate the data masking definition.
    - The supplied algorithm only supports fields defined as strings.
    - Enter `alg="algorithm name"` to reference the masking algorithm. The corresponding Algorithm Type must reference the Data Masking algorithm entity.
    - For data accessed via a scheme-based object call, reference a metadata field name from its schema definition. For example, to mask a credit card number with a schema of `<creditCard mdField="CCNBR" mapField="EXT_ACCT_ID"/>`, set the option value to `field="CCNBR", alg="algorithm name"`.
    - For data accessed through a page maintenance service call, indicate the table name and the field name where the data resides, for example

`table="table_name", field="fld_name", alg="algorithm name"`.

- A WHERE clause may also be specified, which is useful for data that resides in a child table where only data of a certain type needs to be masked. For example, `table="CI_PER_ID", field="PER_ID_NBR", alg="algorithm name", where="ID_TYPE_CD='SSN'"`
- For data stored as a characteristic, indicate the characteristic type as `CHAR_TYPE_CD='char type', alg="algorithm name"`. This needs to be defined only once regardless of which characteristic entity the char type may reside in. Note that only ad-hoc characteristics are supported at the present time.
- For data displayed via a search service call, indicate the search name and the appropriate field to mask along with the masking algorithm. For example, `search="SearchServiceName", field="PER_ID_NBR", where="ID_TYPE_CD='SSN'", alg="algorithm name"`. To find the search service name, launch the search in question, right-click the filter area, select **View Source** and search ServiceName. To find the field name to mask, return to the search window and right-click the search area then select **View Source**. Look for the **Widget Info** section and find the field name in the search results excluding the $. Note that the WHERE statement can only apply to fields that are also part of the search results.

Securing Files

Note: The utilities mentioned in this section apply to Linux and Unix environments only.

The file structure of your application is protected by permissions set at the operating system level. By default, the settings provided upon installation comply with Oracle standards in respect to permissions. For more details of the individual user permissions on product directories and subdirectories, refer to the *Server Administration Guide*.

You can use the following process to reset the environment permissions to the default settings:

1. Execute the `splenviron.sh` utility to set the environment variables for the product environment to reset. Refer to the *Server Administration Guide* for details of this process.
2. Execute the `setpermissions.sh` utility to reset the environment permissions back to the defaults.

Password Management

Your application requires several passwords, which must be changed on a regular basis to conform with security best practices:

Password Owner	Location	Comments
Online User	Domain Authentication Source	No configuration changes. User changes password in security repository directly or indirectly using security products. Security repository is configured in Oracle WebLogic. `WEB_SPLPASS` specifies the default password for the initial user. If this user is used past the installation the password may need to be changed. Refer to the *Server Administration Guide* for more details.
Web Service User	Domain Authentication Source	No configuration changes. User changes password in security repository directly or indirectly using security products. Security repository is configured in Oracle WebLogic.
Batch User	Operating System	No configuration changes. User changes password in security repository directly or indirectly using security products. Security repository is configured in Oracle WebLogic.

Database Users	`BATCH_DBPASS`
	`DBPASS`
	`XAI_DBPASS`
		The database users are stored in `ENVIRON.INI`. Refer to the *Server Administration Guide* on process to change values. New Passwords need to be re-encrypted.

JMX Users	`BSN_JMX_SYSPASS`	The default JMX user is stored in `ENVIRON.INI`. Refer to the *Server Administration Guide* on process to change values. New Passwords need to be re-encrypted.
Administration Account	`WLS_WEB_WLSYSPASS` `WEB_WLSYSPASS`	The default administration users are stored in `ENVIRON.INI`. Refer to the *Server Administration Guide* on process to change values. New Passwords need to be re-encrypted.

Securing Online Debug Mode

Your application features an online debug mode which is used for problem solving and development personnel to trace their code or to diagnose problems. As with other functions within the product the debug function is security-controlled.

To use this facility any of the User Groups an individual user must include *Inquire* access to the `F1DEBUG` Application Service. This will enable the debug facility from the URL.

For more information about the Debug facility refer to the *Server Administration Guide*.

Securing Online Cache Management

The online cache management function resets the online cache to force new values to be loaded. As with other features within the product, it is security-controlled.

To use this facility on any of the User Groups, you must include Change access to the `F1ADMIN` Application Service, which enables the cache management facility from the URL.

For more information about the cache management facility refer to the *Server Administration Guide*.

Web Services Security

Note: This section outlines the Inbound Web Services security facility only.

Note: Refer to Migrating from XAI to IWS (Doc Id: 1644914.1) for more information.

Inbound Web Services allows external web service-based integrations to access functionality within the application. The security settings for the Inbound Web Services can be summarized as follows:

- Inbound Web Services rely on Web Services standards supported by Oracle WebLogic for authentication support.
- Inbound Web Services supports the WS-Policy standards supported by Oracle WebLogic to provide both transport and message security. Refer to the Oracle WebLogic documentation for details of the WS-Policies supported. The following rules apply to those policies:
  - Oracle WebLogic policies are supported if the corresponding setup is performed within Oracle WebLogic. For example, encryption is supported if keystores are configured for encryption keys.
  - WS-Policies are attached within the Oracle WebLogic console or Oracle Fusion Middleware Control after deployment. These policies are maintained independently as per the console documentation.
  - Element Level policies are not supported in the current release.
  - Security policies at the operation level are not supported directly but are supported via authorization.
  - The product ships an internal policy for backward compatibility (UserToken).
- Inbound Web Services uses the underlying business objects, maintenance objects, business services and service scripts to determine authorization of records. This includes authorization for specific operations.
- Inbound Web Services can use Oracle Web Service Manager for additional WS-Policy support and web service access controls.
- Security policies can vary between individual Inbound Web Services.
- Multiple WS-Policies are supported per Web Service. The clients calling these services must conform to at least one of the policies attached.

By default, the WS-Client calling the product must supply an authentication token in the format configured on the WS-Policy on individual web services. By default, there is no default user on Inbound Web Services transactions. A default user may be configured on the `ouaf.ws.defaultUser` setting in the `spl.properties` file for the Inbound Web Services. Refer to the *Server Administration Guide* for details of the process.

> **Note**: Setting of a default user is not recommended for implementations unless backward compatibility is required for older XML Application Interface-based services.

For backward compatibility there are several additional settings that cover Inbound Web Services:

Setting	Comments
`ouaf.ws.defaultUser`	Default user for authorization of Web Services calls
`ouaf.ws.superusers`	Delimited set of effective users to translate calls from authentication users not known to the system.
`ouaf.ws.deploy.user`	Administration user for deployment activities. This setting is only specified if differs from the administration settings.

Message Driven Bean Security

> **Note**: Refer to the Oracle WebLogic JMS documentation for detailed information about JMS facilities provided.

> **Note**: Refer to the Oracle WebLogic JMS Integration (Doc Id: 1308181.1) whitepaper for details of the JMS integration implementation.

The Message Driven Bean (MDB) within the Inbound Web Services implementation allows JMS resources (such as JMS Queues or JMS Topics) to be read using the MDB and sent to an Inbound Web Service to be processed.

By default, the Message Driven Bean, uses the JMS (JMSX) message property fields for authentication and authorization purposes such as `JMSXUserid`.

If the JMS message security is not used then a default user can be set in the `ouaf.ws.defaultUser` parameter in the `spl.properties` file.

SOAP Security

In this release, additional SOAP Header security, for outbound communications, has been added to support additional facilities in Service Oriented Architecture integrations and Oracle Web Services Manager. The following additional facilities are now supported in the SOAP Header:

Facility	Description
SOAP Insert Time Stamp	A set of timestamps can be added to the transaction to support WS-Security to avoid replay attacks.

Facility	Description
Additional SOAP Security inbuilt	The SOAP Header can now have inbuilt support for TEXT and DIGEST headers in addition to the BASIC support provided in past releases. This feature can be replaced using Oracle Web Services Manager with WS-Policy support for other advanced security configurations.
SOAP Expiration Delay	It is possible to set a transaction expiration, in seconds, to control resource usage of the transaction.

Refer to the online documentation for a more detailed description of these settings.

Note: The only HTTP/HTTPS method supported in this release is POST.

Groovy Support

Your application now supports Groovy for extensions, via the script engine. This support was added to augment the Java and Scripting support to offer an alternative. The implementation of Groovy has some limitations for security reasons:

- Groovy API's that have direct access to operating system functions have been blocklisted for security reasons and therefore cannot be used. Alternative functions are provided to offer safe access to selected operating system functions.
- It is possible to implement a custom allow list for non-cloud implementations. Refer to the *Server Administration Guide* for more information.

Refer to the online documentation for more details of Groovy support.

Oracle Cloud Object Storage Support

Note: Prior to using this capability, the Oracle Cloud Object Storage Service must be purchased and configured. Networking between on-premise or other cloud services must be installed, configured and operational before using this facility.

By default, use of `FILE-PATH` batch variable was restricted to local mounted storage where it is possible to use network storage through mapped directories. It is now possible to use Oracle Cloud Object Storage Service as a source of import files or locations to write files.

To use this feature, Oracle recommends the following:

- Create or Edit a lookup value for the `F1-FileStorage` Extendable Lookup for each Cloud Service used with the following Connection Details:

Connection Details	Comments
File Adapter	Use *Oracle Cloud Object Storage.*
Tenancy	OCI Tenancy for Oracle Cloud Object Storage

Connection Details	Comments
Compartment	Object Storage Compartment
Namespace	Object Storage Namespace
Key Ring	Key associated with Oracle Cloud Object Storage
Region	Oracle Cloud Infrastructure Region
Bucket Name Prefix	Prefix to use for Bucket
Reporting Configuration	Whether this setting is for BI Publisher files (Oracle Utilities Cloud Services use only)

- To use the definition, the parameter should be used in the `FILE-PATH` variable, in either the Batch Control definition or batch configuration file for relevant batch controls in the format:

  ```
  file-storage://<ExtendableLookupValue>
  ```

  Where:

  - `<ExtendableLookupValue> is the n`ame of lookup value configure in the `F1-FileStorage` Extendable Lookup

Additional settings may be added to `FILE-PATH` to support virtual folders if necessary.

HTTP Proxy Support

If HTTP Proxies are used for networking these can be configured at the JVM level for all JVMS using the Java Networking and Proxy settings. These settings can be set in the following areas:

- For online, Inbound Web Services, REST and outbound messages, the settings may be specified on the Oracle WebLogic Server settings or using the `GLOBAL_JVMARGS` configuration parameter.
- For Batch in Oracle Coherence, the settings may be specified using the `GLOBAL_JVMARGS` configuration parameter.

Refer to the *Server Administration Guide* for additional information.

SYSUSER Account

The installation of your application supplies an initial account `SYSUSER`, by default. This account is defined by default in the default security realm in the provided templates, is provided as the initial User object in the authorization model and is used as the default user in some transactions.

You cannot physically remove the `SYSUSER` account, as it is used by the initial installation and owned by the application, but it can be disabled under the following conditions:

- Alternative identities have been configured for the authentication and authorization components of the product.

- Every facility in the implementation that uses the SYSUSER account as the default identity has been changed to an alternative to prevent misconfiguration of the facility.

  Note: Oracle recommends that you use the appropriate alternatives for transactions instead of the SYSUSER account.

The following facilities use SYSUSER as the default identity, if used:

Facility	Comments
Default User in Message Option	This is used for XAI and MPL (older releases only).
HTTP Login User context type on Message Sender	Used for Utilities P2P integrations. Ensure HTTP User Password is also set correctly for alternative.
WEB_IWS_SUPER_USERS in ENVIRON.INI file	Used for supporting IWS user proxy connections
WEB_IWS_MDB_RUNAS_USER in ENVIRON.INI file	Used to support user proxy connections in Message Driven Bus.
Batch Controls	Replace SYSUSER as the user used for submission of any batch controls in batch control configuration files, batch edit configuration files or in the Oracle Scheduler configuration.

You can deactivate the SYSUSER account by:

- Removing SYSUSER from configured security realm for authentication, preventing the user from authenticating.
- Setting the **User Enable** attribute (SYSUSER User object) to *Disable*, deactivating the account from any unauthorized activity in your application.

Embedding User Experience

The product supports embedding application and being embedded in applications. To use this facility several settings must be configured appropriately including:

- Set embedded behavior. Set the following values in the spl.properties file used for the online:

Setting	Comments
com.oracle.ouaf.web.csp.enable	Set to true
com.oracle.ouaf.web.allowCORS	Set to true
com.oracle.ouaf.web.csp.allowedFrameAnsMaxNumber	Set to 2 or more
com.oracle.ouaf.web.disableSecureCookie	Set to false
com.oracle.ouaf.uriSubstitutionVariables.file	Set to location of substitution file

- Set the URI substitutions. Set the substitution variables for your embedded or embedding application to avoid hardcoding in the file indicated in com.oracle.ouaf.uriSubstitutionVariables.file. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<substitutionVariables>
…
<uriVariable>
<name>CSP_FRAME_ANS_HOST1</name>
   <value>https://[FRAMING-HOST-URL]</value>
</uriVariable>
<uriVariable>
<name>CSP_FRAME_ANS_HOST2</name>
   <value>https://[FRAMING-HOST-URL]</value>
</uriVariable>
</substitutionVariables>
```

Chapter 8

Audit Facilities

Your application's inbuilt, configurable auditing facility provides the capability to register accesses to data from online and Web Services users. Batch processing is not audited by default but can be enabled using the Oracle Utilities SDK using programmatic methods.

Auditing allows for the configurable tracking of changes to key data and allows authorized users to track changes on an individual user. Use of this facility is optional and can be switched on or off at any time.

The following topics are included:

- Audit Configuration
- Audit Query by Table/Field/Key
- Audit Query By User
- Read Auditing
- Integrating to Audit Vault

Audit Configuration

Note: This section covers the **soft-table implementation** of auditing. There is a specialist Audit algorithm support on Business and Maintenance objects to add information to log entries attached to these objects. Refer to the Oracle Utilities SDK and online Administration documentation for a description of programmatic implementation of auditing.

Audit configuration for Oracle Utilities Application Framework is performed at the table level. Enable auditing on each table by navigating to the **Administration** menu then the **Table** menu option, and configuring the following field settings:

- **Audit Table:** You need to configure a database table to store the audit information. By default, the CI_AUDIT table can be used for this purpose. When using a custom table, make sure that the structure of this table is similar to CI_AUDIT to ensure compatibility.
- **Audit Program:** You must configure a class or program that will record and process the audit information. By default, several pre-built audit programs are available for use:
  - `com.splwg.base.domain.common.audit.DefaultTableAuditor` – This is the default java-based audit class provided by the product. It audits any changes to fields configured to track auditing information.
  - `com.splwg.base.domain.common.audit.ModifiedTableAuditor` – This is an alternative to the DefaultTableAuditor class. However, it does not audit inserts or deletions of empty string field data. For example, changes from null values to empty spaces or empty spaces to null values are not logged.

  Note: It is possible to implement custom Audit handlers using the base classes as parent classes. Refer to the Oracle Utilities SDK documentation on how to extend the product.
- **Audit conditions:** A set of switches are configurable on each field you wish to include in auditing to determine the conditions of auditing. At least one of these switches must be enabled for auditing to be registered:
  - **Audit Delete Switch:** Enable this switch to audit delete operations against this field.
  - **Audit Insert Switch:** Enable this switch to audit insert operations against this field.
  - **Audit Update Switch:** Enable this switch to audit update operations against this field.

To maintain the audit information, navigate to the Table menu option on the Administration menu and specify the table to enable auditing against.

Specify the Audit Table, Audit Program (and associated type) and configure the Audit Switches on the fields you wish to track.

Note: To enable Auditing on a running version of the product, the online data cache must be flushed, or the product restarted. Refer to the Server Administration Guide for more details.

Audit Query by Table/Field/Key

Once Auditing is activated, changes are logged in the configured Audit Table using the Audit Program specified in the configuration. It is possible to query this Audit information by Table, Field and Key value to isolate changes. To access this query, navigate to the Audit **Query By Table/Field/Key** menu option on the Administration menu.

Specify any the following values for the filters:

- **Audit Table Name:** The name of the table that has been audited to query. When this table is chosen the screen will list additional fields to filter upon.
- **Audit Field Name:** The name of the field to track to filter the results.
- **Creation Start Date/Time and Creation End Date/Time:** Date and time range to limit the records returned.

The query will return the following results:

- **Create Date/Time:** Date and time the changes were made.
- **User Name:** Name of user who made the changes.
- **Primary Key:** Record key of the change.
- **Audited Field Name:** Name of field that was changed.
- **Audit Action:** Action that was recorded with change (i.e. Insert, Update or Delete)
- **Value Before Audit**: The field value before the change was made.
- **Value After Audit:** The field value after the change was made.

Audit Query By User

Once Auditing is enabled changes are logged in the configured Audit Table using the Audit Program specified in the configuration. It is possible to query this Audit information by individual users to isolate changes made by that user. To access this query, navigate to the **Audit Query By User** menu option on the **Administration** menu.

Specify any the following values for the filters:

Value	Description
User ID	Authorization User to track.
Audit Table	The name of the table containing the audit information.
Creation Start Date/ Time and Creation End Date/Time	Date and time range to limit the records returned.

The query will return the following results:

Result	Description
Row Creation Date	Date and time the changes were made.

Result	Description
Audited Table Name	Name of table that was audited.
Primary Key	Record key of the change.
Audited Field Name	Name of field that was changed.
Audit Action	Action that was recorded with change (Insert, Update or Delete).
Field Value Before Audit	The field value before the change was made.
Field Value After Audit	The field value after the change was made.

Read Auditing

Whilst the inbuilt Audit facility is mainly used to register changes in data, it can also be used to register whenever data is accessed for auditing purposes. Read Auditing is different from the standard auditing as it focuses on zones. On the zone configuration there is an ability to configure an Audit Service Script which is called whenever the zone is displayed to determine which criteria and result records are displayed. In the current release, Read Auditing is available for the following zone types:

- F1-DE
- F1-DE-QUERY
- F1-DE-SINGLE
- F1-MAPDERV
- F1-MAPEXPL

The information audited can be determined by using programs and logged based on your requirements. Refer to the online help for descriptions and samples for Read Auditing.

> **Note**: Products ship with sample generic inquiry Audit code specific to the product. These can be reused or altered to suit your needs. Refer to the product documentation for details of these samples.

Integrating to Audit Vault

> **Note**: Customers using Oracle 12.x and above, should use Unified Auditing to ensure consistent capture of audit information.

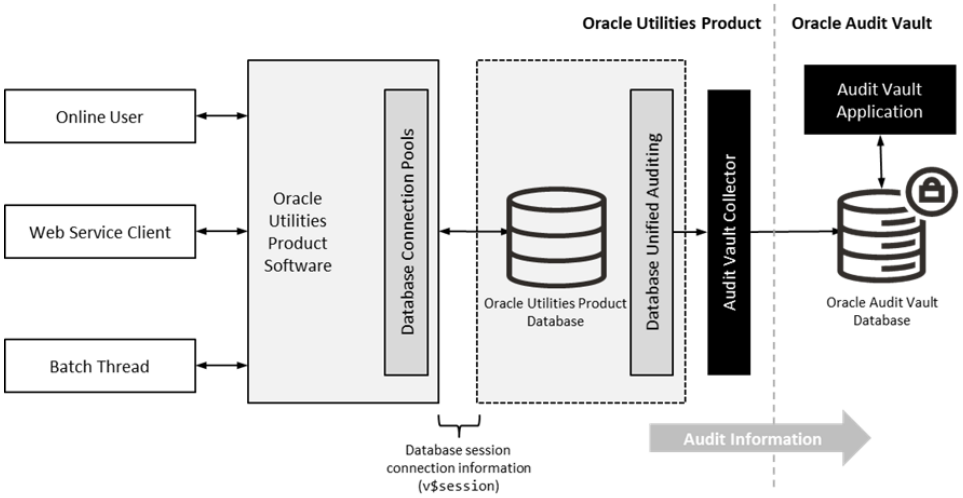
The Oracle Utilities Application Framework contains an internal audit facility that provides a basic audit facility for recording changes and optionally, inquiring, data by online users. Whilst this facility is enough for most needs it may be replaced with using Oracle Audit Vault to provide an enterprise wide audit facility.

Oracle Utilities Application Framework supports the use of Audit vault in association or as a complete replacement for the inbuilt Auditing feature.

Audit Vault collects audit information at the database level, using the Database Auditing features of the Oracle Database, and loads them into a separate Audit Vault database.

The information in that database can be queried, reported and managed using the Audit Vault front end.

For example:



To use Audit Vault, the following must be configured:

Configuration	Description
Setup Database Auditing	The Database auditing feature must be enabled to store the relevant audit information. The level of auditing information and the location of the audit information is configurable. Refer to the Oracle Database Security Guide for a discussion of Database Auditing and Best Practices of Auditing for a discussion of the various methods available.
Design Database Auditing	The tables, users and SQL statements to audit need to be specified on the product database. This typically done by the database administrator using the AUDIT statement.
Install and Configure Audit Collector	On the host holding the product database an Audit Vault Agent needs to be configured to pass audit information to Audit Vault and implement data retention policies for audit information.
Configure Audit Vault	Audit Vault can be configured to implement policies, alerts and reports on the Audit data. Audit Vault can be configured to set an Audit Data Retention Policy for its internal audit information.

Chapter 9

Database Security

The Oracle Database supports a wide range of security configurations natively or via additional options available. For a full discussion of the available security options for the database refer to the Oracle Database Security Guide.

The following topics are included:

- Database Users
- Database Roles
- Database Permissions
- Using Transparent Data Encryption
- Using Database Vault

Database Users

Your application installation ships with a predefined set of users to be used by the product at configuration and runtime. These users are specified in the installation of the product to build the database and load its initial dataset.

The following users are available:

- SPLADM – This is the default DBA administration account which owns the product schema. This user is used to create and maintain the structures of the database. It is used by DBA personnel to maintain the product schema and indexes.
- SPLUSER – This is the default main product user used by the product to access the SPLADM schema. The product uses this physical userid as a pooled user with pooled connections to the database. Variations on this account can be created for each channel of access using the following configuration settings.

Configuration Parameter	Comments
BATCH_DBUSER	Database User for Batch
DBUSER	Database User for online (Default: SPLUSER)
XAI_DBUSER	Database User for Web Services

- SPLREAD – This is the default read only user available for reporting tools or external direct interfaces to use on the product database. This user is not used by the product.

  Note: For customers on older versions of products this user was also used for the ConfigLab component.

- CISOPR, OPRPLUS – These are optional operator users that can be used to delegate backup and restore operations on the product.

  Note: The values of these users can be altered to customer specific values at installation time. Refer to the product Installation Guide and product DBA Guide for more information.

Database Roles

Your application ships with a set of database roles to allow administrators to allocate new database users to the relevant components of the product. The following roles are shipped by default for the product:

- SPL_USER – This role is available for database users who require update, insert, delete and select access to the product schema. This role is used for product users.
- SPL_READ – This role is available for database users who require read only access to the product schema.

To use the roles the DBA grants the role to the database user to connect them to the schema in the desired fashion.

Database Permissions

Database permissions for the application are allocated at the role level with the role setting permissions to the schema objects. By default, the roles have full access to all the objects in the product schema, as dictated by the role.

Unless otherwise stated, it is not recommended to alter the database users used by the product to specific additional permissions on the product schema as this may cause permission issues.

Customers wishing to restrict external parties, such as external tools or reporting engines, to specific objects may use all the desired security facilities available in the database to implement those restrictions.

Using Transparent Data Encryption

Transparent Data Encryption (TDE) allows data to be encrypted at the storage level to protect the data files at the lowest level. From a product perspective, the implementation of Transparent Data Encryption requires no product configuration changes on the application server.

> **Note**: To implement Transparent Data Encryption, DBAs will have to execute appropriate alter statements on product tables to indicate the level of encryption.

For more information about implementing Transparent Data Encryption refer to the Oracle Advanced Security Guide.

Using Database Vault

By default, the database administration account as SQL Data Manipulation Language (DML) access to the application's schema, as dictated by the default permissions of the Oracle Database. It is possible to restrict the permissions of the DBA to SQL Data Definition Language (DDL) statements only using Database Vault. Refer to the Database Vault Administrators Guide for details of this facility.

The product includes a prebuilt database vault solution, refer to the Database Vault Integration (Doc Id: 1290700.1) available from My Oracle Support.

Chapter 10

Security Integration

Whilst Oracle Utilities Application Framework applications provide a set of security facilities natively or via Oracle WebLogic, it is possible to augment the security with additional security features or security products.

The following topics are included:

- LDAP Integration
- Single Sign On Integration
- Oracle Identity Management Suite Integration
- OAuth2 Support

LDAP Integration

By default, Oracle WebLogic includes an internal security repository that uses the Lightweight Directory Access Protocol (LDAP) to provide authentication facilities.

> **Note**: It also provides authorization services, but these are not typically utilized by the product.

It is possible to replace the internal security repository with another LDAP compliant security source.

To use an alternative source as a security repository the following process must be used:

Oracle WebLogic must be configured to use the external LDAP security source for authentication. Refer to the documentation provided with Oracle WebLogic for more details. For Oracle WebLogic customers, refer to the Configuring LDAP Authentication Providers section of the Oracle Fusion Middleware Securing Oracle WebLogic Server Guide.

The product LDAP import feature can be used to initially populate the authorization model from the LDAP source as outlined in the LDAP Integration for Oracle Utilities Application Framework based product (Doc Id: 774783.1) available from My Oracle Support.

> **Note**: Whilst LDAP sources are the most common security repository, it is possible to use alternative security authentication sources as supported by Oracle WebLogic. Refer to the documentation provided with Oracle WebLogic for more details.

Single Sign On Integration

One of the common security integrations is the ability to implement Single Sign On with your application. This enables end users to access the product minimizing the need to re-authenticate each time.

Oracle WebLogic in association with other technologies can be configured to support Single Sign On. For more details refer to Single Sign On Integration for Oracle Utilities Application Framework based products (Doc Id: 799912.1) and Oracle Identity Management Suite Integration with Oracle Utilities Application Framework based products (Doc Id: 1375600.1) available from My Oracle Support.

Kerberos Support

Single Sign-On (SSO) with Microsoft clients allows cross-platform authentication between Web applications running in Oracle WebLogic and .NET Web service clients or browser clients (for example, Microsoft Internet Explorer) in a Microsoft domain. The Microsoft clients must use Windows authentication based on the Simple and Protected Negotiate (SPNEGO) mechanism.

Refer to Configuring Single Sign-On with Microsoft Clients for details of configuring Oracle WebLogic to use Kerberos.

Oracle Identity Management Suite Integration

Oracle offers a comprehensive set of security products as part of the Oracle Identity Management Suite that can be used to augment the security setup at your site. The product can be integrated with the following components of Oracle Identity Management Suite:

Component	Description
Oracle Identity Manager	Oracle Identity Manager can be used to centralize user provisioning to the product, password rule management and identity administration.
Oracle Access Manager	Oracle Access Manager can be used to provide authentication, single sign on, access controls and user tracking.
Oracle Adaptive Access Manager	Oracle Adaptive Access Manager can be used to provide fraud tracking and multi-faceted authentication.
Oracle Virtual Directory	Oracle Virtual Directory can be used to provide virtualized LDAP security access to LDAP and non-LDAP security sources.
Oracle Internet Directory	Oracle Internet Directory can be used as a LDAP security store.

Refer to the Oracle Identity Management Suite Integration with Oracle Utilities Application Framework based products (Doc Id: 1375600.1) whitepaper for more information, available from My Oracle Support.

OAuth2 Support

Note: For customers using OAuth2 for Federated Security, refer to the dedicated section in this guide for detailed instructions.

OAuth2 is supported for delegated access to API's within the product. From the configuration perspective the following is required for Oauth2:

- The `CLIENT-CERT` setting should be specified for security in the product. This indicates that the security is coming from an external source to the domain.
- Configure Oracle WebLogic domain security realm to support OAuth2 using the instructions provided with the relevant version of Oracle WebLogic.
- If you are using a service provider like Oracle Identity Suite or Oracle Access Manager in association with Oracle WebLogic, then refer to the documentation provided with those products for domain specific setup instructions.
- If you are using a third-party provider in association with Oracle WebLogic, refer to the documentation provided with that provider with additional setup instructions.

Chapter 11

Keystore and Truststore Support

Oracle Utilities Application Framework applications support the ability to store cryptographic keys and certificates. Keystores are used to encrypt and decrypt data, such as passwords, and for the data encryption component of the Oracle Utilities Application Framework. Because of their importance, it is vital that keystore-related files be secured and only accessible by administrators. It is also possible to implement a truststore to ensure the integrity of certificates.

The following topics are included:

Creating the Keystore and Truststore

Note: For backward compatibility, customers on older versions will have a default keystore and truststore created upon upgrade with backward compatible values.

Note: If the keystore and truststore is not present, Oracle Utilities Application Framework will revert to the internal cryptography used in previous releases.

Note: Passwords encrypted using this keystore will be prefixed with ENCKS and legacy password encryption uses prefix ENC.

Note: The settings in this section are defaulted in installation and can be altered using overrides in `templates FW_spl.properties.keystore.truststore.include` as outlined in the *Server Administration Guide*.

Typically, a keystore and truststore are created using the java keytool utility manually but the Oracle Utilities Application Framework utilities have been extended to allow customers to create and manage the keystore from the command line.

Before creating the keystore the following settings must be set in the installation, as per the Server Administration Guide:

- `KS_ALIAS` - The alias used to encrypt/decrypt passwords by the Oracle Utilities Application Framework to access the keystore. By default, this is set to `ouaf.system`.
- `KS_ALIAS_KEYALG` - The algorithm to be used by the `KS_ALIAS` entry in keystore to encrypt the passwords. By default, this is set to `AES`.
- `KS_ALIAS_KEYSIZE` - The strength of the keystore for the `KS_ALIAS` entry. By default, this is set to 256.
- `KS_HMAC_ALIAS` - The HMAC alias used by the Encryption Feature Type of the Oracle Utilities Application Framework. By default this is set to `ouaf.system.hmac.`
- `KS_HMAC_ALIAS_KEYALG` - The algorithm to be used by the `KS_HMAC_ALIAS` entry in keystore to encrypt the data. By default, this is set to `HmacSHA256`.
- `KS_HMAC_ALIAS_KEYSIZE` - The strength of the keystore for the `KS_HMAC_ALIAS` entry. By default, this is set to `256`.
- `KS_KEYSTORE_FILE` - Location of the keystore file.
- `KS_MODE` - Keystore Padding mode. By default, this is set to `CBC`.
- `KS_PADDING` - Key padding algorithm used for keystore. By default, this is set to `PKCS5Padding`.
- `KS_STOREPASS_FILE` - Keystore Password file.
- `KS_STORETYPE` - Keystore type. By default, this is set to `JCEKS`.
- `TS_ALIAS` – Alias used for trust store. By default this is set to `ouaf.system`
- `TS_ALIAS_KEYALG` - The algorithm to be used by the `TS_ALIAS` entry in truststore. By default, this is set to `AES`.
- `TS_ALIAS_KEYSIZE` - The strength of the truststore for the `TS_ALIAS` entry. By default, this is set to `256`.

- TS_HMAC_ALIAS - The HMAC alias used by the truststore. By default, this is set to ouaf.system.hmac.
- TS_HMAC_ALIAS_KEYALG - The algorithm to be used by the TS_HMAC_ALIAS entry in truststore to encrypt the data. By default, this is set to HmacSHA256.
- TS_HMAC_ALIAS_KEYSIZE - The strength of the truststore for the TS_HMAC_ALIAS entry. By default, this is set to 256.
- TS_KEYSTORE_FILE - Location of the truststore file.
- TS_MODE - Truststore Padding mode. By default, this is set to CBC.
- TS_PADDING - Key padding algorithm used for truststore. By default, this is set to PKCS5Padding.
- TS_STOREPASS_FILE - Truststore Password file.
- TS_STORETYPE - Truststore type. By default, this is set to JCEKS.

Once these settings are specified the keystore/truststore is created using the following command:

```
initialSetup.sh -k|-K
```

This generates the keystore (-k) or truststore (-K) using the credentials outlined in the Keystore or Truststore Password file.

Altering the KeyStore/Truststore options

Note: This process should be used for any keystore/truststore change including copying keystores/truststores across environments.

After creating the keystore if any of the keystore values need to be changed then the system needs to be realigned to the new configuration. The following process must be performed:

- Logon to the machine where you wish to make the changes to the settings.
- Execute the splenviron[.sh] -e <environment> command where <environment> is the environment on the machine to change.
- Shutdown the environment.
- Alter the keystore parameters to suit the new desired configuration using the configureEnv[.sh] -a utility.
- Execute the initialSetup[.sh] -k utility to recreate the keystore (-k) or initialSetup[.sh] -K to recreate the truststore (-K) with the new settings.
- Execute the configureEnv[.sh] once more and press enter on each password prompt to re-encrypt the passwords with the new settings.
- Execute the initialSetup[.sh] command to apply the changes to the configuration files.

  Note: For customers using native installation, update the Deployments using the Oracle WebLogic console or Oracle Enterprise Manager to load the new versions of the product EAR files.

- If the encryption values have changed the data encrypted in the database must be re-encrypted to match the new settings using the process outlined in Synchronize Data Encryption.

Synchronize Data Encryption

Note: Failure to synchronize data when encryption values change will cause outages and unexpected behavior in the application.

Note: Shut down your application while running this process.

If at any time the encryption values change the values that are encrypted using the old value must be updated to reflect the new settings. A new utility `com.splwg.shared.common.ChangeCryptographyKey` is provided to synchronize data changes. The following keys are updated using this utility:

- Database Passwords used in Feature configurations such as Database Update features.
- Message Sender and Receiver Passwords (depending on Sender and Receiver type)
- Reporting tool integration passwords
- Multi-Purpose Listener passwords (for selected products)
- Email Adapter configuration.
- Web Services Passwords (legacy only)
- Security Hashes on user records

The following process is to be used:

- Logon to the machine you have made the changes upon as the product administrator.
- If you have not already done so, use the `splenviron` utility to set the environment variables for the product environment.
- Execute the following command:

```
perl <SPLEBASE>/run_java_standalone.plx
com.splwg.shared.common.ChangeCryptographyKey[-t|-l|-h|-p]
[old-settings]
```

where options are:

`-t`	Test Mode (no commit of changes)
`-l`	Convert Legacy/OUAF System key
`-h`	Convert User hashes only
`-p`	Convert encrypted passwords only
`[old settings]`	List of old settings as per below (other above options should not be used with these settings)

```
-Dcom.oracle.ouaf.system.old;.keystore.file=<oldfile> -
Dcom.oracle.ouaf.system.old.keystore.passwordFileName=<oldpassf
ile> -Dcom.oacle.ouaf.system.old.keystore.type=<oldtype> -
Dcom.oacle.ouaf.system.old.keystore.alias=<oldalias> -
Dcom.oacle.ouaf.system.old.keystore.padding=<oldpadding> -
Dcom.oracle.ouaf.system.old.keystore.mode=<oldmode>
```

Where:

<oldfile>	Original Key Store file
<oldpassfile>	Original Password Store file
<oldtype>	Original Key store type
<oldalias>	Original alias
<oldpadding>	Original Padding
<oldmode>	Original Mode

Note: Only specify the values that have been changed.

Note: This command must be run once for each alias.

After running `ChangeCryptographyKey`, you must run `$SPLEBASE/bin/invokeDBUpdatePatch.sh` to reset the database patching credentials as follows:

- If you have not already done so, use the `splenviron.sh` utility to set the environment variables for the product environment.
- Run the command with the `-b` option to go into interactive mode and reply to the prompts. Use the `-h` option to get help.

```
$SPLEBASE/bin/invokeDBUpdatePatch.sh -b
```

Upgrading from Legacy to Keystore

When upgrading from past releases of Oracle Utilities Application Framework and adopting the new keystore it is recommended to use the following process to adopt the keystore:

- Ensure all passwords have been updated by executing the `configureEnv` and pressing enter at each password prompt.
- Execute the process outlined in Synchronize Data Encryption running the `com.splwg.shared.common.ChangeCryptographyKey` utility with the `-l` option to convert old keys to new keys. For example:

  ```
  java ChangeCryptographyKey -l
  ```

- Ensure that you also execute the `invokeDBUpdatePatch.sh` process outlined in Synchronize Data Encryption.
- Optionally, it is possible to update the passwords using the `LegacyCryptographerUpdater` utility on individual passwords using the following command:

  ```
  java LegacyCryptographyUpgrader [-f <file>| -p <password>]
  ```

  where options are:

`-f` *<file>*	Read *<file>* for password and re-encrypt to stdout
`-p` *<password>*	Decrypt old password *<password>* and re-encrypt to stdout. The password should be already in ENC format.

For more information and examples, refer to the Installation Guide and to Oracle Utilities Application Framework - Keystore Configuration (Doc Id: 2014161.1).

Importing Keystores/Truststores

While your application supplies a default keystore and truststore, it is possible to import existing keystores and truststores from alternative sources (such as a corporate level set of stores or from a trusted CA authority).

To import a keystore or truststore the following process should be followed:

- Logon to the machine you have made the changes upon as the product administrator.
- If you have not already done so, use the `splenviron` utility to set the environment variables for the product environment.
- Ensure the `KS_IMPORT_KEYSTORE_FOLDER` or `TS_IMPORT_KEYSTORE_FOLDER` is set in the `ENVIRON.INI` prior to continuing. These are the locations the files to be imported will be located for keystores and truststores respectively.
- Copy the new keystore or truststore to the locations specified in the `KS_IMPORT_KEYSTORE_FOLDER` or `TS_IMPORT_KEYSTORE_FOLDER` respectively.
- Execute the `initialSetup.sh -s | -S` to import the keystore (`-s`) or truststore (`-S`) successfully.

Chapter 12

Encryption Feature Type

One of the major features of Oracle Utilities Application Framework applications is the ability to mask and encrypt data to protect sensitive information. This encryption is implemented in a Feature Configuration using the Encrypted Feature Type.

The Oracle Utilities Application Framework supports Feature Configuration which store specific configuration settings for features in the product to be implemented. Feature Configurations allow simple configurations to be implemented for specific features.

Feature Configurations can be maintained using the **Feature Configuration** menu option on the **Administration** menu.

For the Encryption feature, one Feature Configuration should exist for the *Encryption* Feature Type with an option per field to encrypt.

> **Note:** If the product does not ship a Feature Configuration for Encryption, then it can be created as a Customer Modification. Prefix the name of the Feature Name with CM.

Configuration of Encrypted Fields

To define a field to encrypt an option must be added with the following attributes:

- Option Type should be set to `Field Encryption`.
- Sequence should be an appropriate sequence number. Typically, this is a number that is not used already. Higher number values override lower level sequences.
- In the value you need to specify the specification of the encryption in the format of a command string.

`table`	Table Name. Table must exist in meta data.	`table="SC_USER"`
`field`	Field to encrypt. Field must exist in metadata.	`field="FIRST_NAME"`
`alias`	Keystore alias to use to encrypt the data	`alias="ouaf.system"`

where	Filter for data. Useful for child tables to determine specific values to encrypt	where="ID_TYPE_CD='SSN'"
wrap	Whether the value should wrapper with the ENC() marker. [true|false]	wrap=false
maskAlg	If the field is also to be masked, then the algorithm to mask the data.	maskAlg="CMCCR"
maskField	If the field is also to be masked, then the field to use as the mask	maskField="CNBR_MASK"
hashAlias	If the field should be hashed, then the alias in the keystore to use	hashAlias="ouaf.hmac.system"
hashField	If the field should be hashed, then the field to use as the hash value	hashField="CNBR_HASH"
encryptedField	If the output from the encryption is to be stored on another field in the table, specify the field name.	encryptedField="PK_VAL2"

For example:

```
table="F1_ATTACHMENT",field="PK_VAL5",alias="ouaf.system",encrypte
dField="PK_VAL2",hashAlias='HmacSHA256-1024',hashField="PK_VAL3",
where="PK_VAL1='Encrypted'"
```

There are a few guidelines when using this facility:

- The aliases specified in alias and/or hashAlias must exist in the keystore used for the product.
- Fields to be encrypted must be in string format only. Other field formats are not supported.
- If using a higher level of encryption may increase the storage requirements for a field. If this is the case, adding an encryptedField to hold the larger encrypted value.
- The wrap field should be set to false unless additional processing in your code is included to handle the special marker. Product fields should use wrap=false. Wrapping an encrypted value can be useful in knowing whether a specific data is encrypted in cases where only some data on the table is encrypted.
- Ad-hoc characteristics cannot be specified in the WHERE tag.
- Hashing the value is handy for additional verification and indexing values.

  Note: If encryption is added or changed, the F1-ENCRS and/or F1-ENCRT must be executed to reflect the changes.

Chapter 13

Web Services Security

In the product the Inbound Web Services capability, based upon a JAX-WS/JAX-RS implementation, allows for support for WS-Security and WS-Policy support on individual Inbound Web Services from several perspectives.

> **Note**: Refer to the Web Services Best Practices (Doc Id: 2214375.1) for additional implementation advice for web services security.
>
> **Note**: This section applies to REST and SOAP based services defined as Inbound Web Services.

The following topics are included:

- Annotation Security
- Oracle WebLogic WS-Policy Support
- Oracle Web Services Manager Support
- Support for Multiple Policies
- Importing Certificates for Inbound Web Services

Annotation Security

It is possible to implement custom WS-Policy support using inbuilt annotation support on individual Inbound Web Service definitions. This allows the policies to be implemented within the product and allows for backward compatibility for sites using XML Application Integration (XAI).

The product supports custom WS-Policy files using the following method:

- The WS-Policy formatted XML file containing the policy definition is installed in the `$SPLEBASE/splapp/iws/resources/policies` directory (or `%SPLEBASE%\splapp\iws\resources\policies` directory on Windows).
- A Web Service Annotation of type F1POLICY is defined using the Web Service Annotation maintenance function. In that Annotation entry the following should be configured:
  - **uri**: The name of the policy XML file located in the policies directory in the format `"policy:<policyname>"` where *<policyname>* is the name of the file containing the policy. For example: **`"policy:UsernameToken.xml"`**.

- **attachToWsdl**: Whether the WS-Policy file is attached to the WSDL (for SOAP Web Services only).
- **direction**: Specifies when to apply the policy as per `weblogic.jws.Policy.Direction`. For example: `Direction.both`

- The annotation is attached to the relevant Inbound Web Services to implement the policy using the Inbound Web Services maintenance function.

  Note: Multiple policies can be added as documented in Support for Multiple Policies.

Oracle WebLogic WS-Policy Support

Note: Customers wishing to use Oracle WebLogic policies should NOT configure policies via the annotations on the same web service. Use of Oracle WebLogic policies and annotations are mutually exclusive and can cause unintentional security violations.

Oracle WebLogic has inbuilt WS-Policy Message-Level and Transport-Level support with predefined policies that can be used with Inbound Web Services at the container level. These policies can be attached within the container on individual services using the Oracle WebLogic console using Attach a WS-Policy to a Web Service using the policy type of WebLogic Web Service Policy.

Note: Message level policy support is restricted to the whole message not within parts of the message.

Multiple policies can be supported directly by specifying additional policies in the Chosen Inbound Message Policies.

Oracle Web Services Manager Support

Note: Customers using annotations can also use Oracle Web Services Manager policies by specifying the F1-OWSM Policy type as an annotation.

Note: Customers wanting to use Oracle Web Services Manager within the product domain should use the Full JRF Profile not the Restricted JRF Profile as recommended as a minimum for the product installation.

Oracle Web Services Manager can be used to secure individual Inbound Web Services providing additional WS-Policies and access control support. As with Oracle WebLogic WS-Policy support, configuration is performed using the Oracle WebLogic console by specifying individual policies on individual Inbound Web Services by using the policy type of Oracle Web Services Manager (OWSM).

Refer to Using Oracle Web Service Manager Security Policies for additional information about Oracle Web Services Manager.

Access Control Support

Note: By default, all Inbound Web Services are accessible by all valid users.

Oracle Web Services Manager allows for access controls to be configured for Inbound Web Services to provide additional security access controls. This facility allows for multiple rules to be configured implementing access rules across the following areas:

- **Basic Policies**: Policies relating to identity, group, role, environment mode and generic global rules.
- **Date and Time Policies**: Policies relating to specific dates and times including periods of access.
- **Context Element Policies**: Policies relating to data within the service itself.

Policies can be individually specified per service operation or combined to implement complex access requirements.

Refer to Security Policy Conditions for more information about specific policies and how to enable policies on services.

Support for Multiple Policies

Within each policy regime (annotation, Oracle WebLogic or Oracle Web Services Manager) it is possible to configure multiple policies where web service clients must conform to at least one policy specified. The following methods are supported:

- **Annotations**: On the Inbound Web Service each policy can be added as an individual annotation with the sequence number designating the order they are checked. It is also possible to delegate Oracle Web Services Manager for policies as part of an annotation using the `F1-OWSM` annotation type.
- **Oracle WebLogic/Oracle Web Services Manager**: In the web service deployment, it is possible to designate multiple policies in the Chosen Policies column of the individual web service WS-Policy configuration. The order of the policies is dictated by the position in the list of Chosen Policies.

Importing Certificates for Inbound Web Services

If your implementation uses certificates for security, the certificate must be deployed with the Inbound Web Services deployment. To perform this activity:

- Ensure the certificate is valid and is installed on the Oracle WebLogic servers used for deployment of the Inbound Web Services.
- Logon to the machine you have made the changes upon as the product administrator.
- If you have not already done so, use the `splenviron` utility to set the environment variables for the product environment.
- Execute the `initialSetup[.sh] -i [<host>:<port>]` where to import the certificate into the Inbound Web Service deployment.

Chapter 14

Allowlist Support

In the Oracle Cloud implementations of your application, the use of allow list is enforced to protect resources within the implementation. These allow lists can also apply to non-cloud implementations and in some cases can be extended to suit individual needs.

> **Note:** Custom allow lists are not supported on Oracle Utilities SaaS Cloud Services.

The following topics are included:

- URL Allow List
- SQL Allow List
- HTML Allow List
- Groovy Allow List

URL Allow List

Note: The URL allow list is blank by default unless otherwise configured as part of your customizations.

Note: For this facility to be used the `com.oracle.ouaf.uriValidation.enable` parameter in the `spl.properties` file must be set to `true`.

It is possible to limit the values of URL's within the product for key objects with the configuration. This is implemented as a allow list that can filter on scheme (aka protocol), hosts and ports. These are checked at runtime and can generate an error if they do not adhere to the allow list.

The feature allows for the following:

- Individual scheme hosts and port combinations can be configured to limit runtime access for specific features.
- Specification of the '*' wildcard is supported for scheme, hosts and ports.

The allow list is configured using the following settings in the `spl.properties` file:

Configuration Parameter	Comments
`com.oracle.ouaf.uriValidation.enable`	Enable or disable URI validation.
`com.oracle.ouaf.whitelist.file`	Location and name of product allow list file.
`com.oracle.ouaf.customer.whitelist.file`	Location and name of custom allow list file.

The format of the allow list file is as follows:

XML Tag	Comments
`"<Parameter>"`	Feature within product to limit. This is a preset string linked to a URI parameter in the product.
`uri`	URI tag.
`scheme`	Protocol supported by *`<Parameter>`*. Valid values will vary depending on the *`<Parameter>`* value. For example, URL's support `file, http, https`, and so on.
`host`	Host name(s) or IP Address(es) to filter upon.
`port`	Port number(s) to filter upon.

For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<whitelist>
…
   "Message Sender HTTP URL properties"
   <uri>
       <scheme>https<scheme/>
```

```
        <host>myhost.mycompany.com<host/>
        <port>*<port/>
    </uri>
…
</whitelist>
```

Refer to the *Server Administration Guide* for more details of the usage and format of this file.

Implementing a Custom URL Allow Lists

It is possible to establish a custom allow list to implement URL allow lists for non-cloud implementations. The format of the allow list is the same as the above allow list and this will augment and enhance the existing allow list (if present). To support a custom allow list perform the following steps:

- Logon to the machine you have made the changes upon as the product administrator.
- If you have not already done so, use the `splenviron` utility to set the environment variables for the product environment.
- If it does not already exist, clone the product allow list located as indicated in the `com.oracle.ouaf.whitelist.file` parameter. Ensure the custom file is in the same location as this allow list file and is prefixed with `"cm"` to indicate it is a customization. The file name can be valid for your operating system and site preferences and must be suffixed with `".xml"`. Uncomment any section you want to set.
- Create a custom `spl.properties` template or use the templating function to set the following parameter with the filename and location of the custom allow list.

  `com.oracle.ouaf.customer.whitelist.file=<filename>`
- Execute the `initialSetup.sh` to implement the new setting.

SQL Allow List

Note: Custom SQL Allow List are not supported at this time.

Sites not wanting to use this SQL function allow list should ensure that the `spl.runtime.customSQLSecurity` is set to **`false`** in the `spl.properties` file.

The SQL used in query zones and in Groovy scripts can be limited in terms of what SQL functions are supported to prevent performance issues or inappropriate access to the database via functions.

The product provided allow list is implemented as a Managed Content object named `F1-SQLFunctionWhiteList`. This allow list is not changeable and lists the supported functions that are allow listed for use. Any function used that is not used in this allow list, if the facility is enabled, will generate a runtime error when the SQL is executed.

HTML Allow List

The HTML used in UI Maps can be limited using a HTML allow list. This product managed allow list lists the valid HTML tags that can be used in the HTML objects. The allow list is implemented as a Managed Content allow list named `F1-HTMLWhiteList`.

Any attempt to run a UI Map with a tag that is not listed in the `F1-HTMLWhiteList` will be ignored (as comments) which may lead to unexpected behavior.

Implementing a Custom HTML Allow List

It is possible to replace the inbuilt HTML allow list by creating a custom HTML allow list to extend the tags supported at runtime in UI Maps.

To implement a custom HTML allow list, copy the `F1-HTMLWhiteList` Managed Content to `CM-HTMLWhiteList` Managed Content and extend the allow list.

> **Note:** Do not remove any tags that exist within `F1-HTMLWhiteList` in your custom allow list. This may cause unexpected behaviour across base UI Maps.
>
> Any upgrades to `F1-HTMLWhiteList` must be also manually reflected in `CM-HTMLWhiteList` for any subsequent upgrade.

Groovy Allow List

> **Note:** The current implementation of the Groovy Allow List is dynamically generated and cannot be altered.

The Groovy language has been added as an alternative language used for scripting. As the Groovy language can access low level API's it has been allow listed to exclude parts of the language not appropriate for cloud implementations. The allow list confirms to the Oracle Cloud SDK Supported Groovy Classes and Methods. ADF extensions to Groovy are not supported. Refer to the online documentation for additional advice and examples.

Chapter 15

Custom Authentication Service Provider

In the Oracle Utilities Application Framework, a custom Oracle WebLogic Authentication Service Provider has been provided to support complex domain security configurations.

The following are included:

- What does this Security Provider do?
- Where would I use this Security Provider?
- Implementing the Security Provider

What does this Security Provider do?

The Oracle Utilities Application Framework Service Provider allows Oracle WebLogic domains to be extended with the following additional checks:

- The provider will check that the authentication user has been defined as a User object within the product. If the user does not exist, then the provider will issue an error to be processed according to the rules in the domain security setup (including using the Oracle WebLogic Adjudicator Providers to implement access rules).
- If the user exists as a User object within the product, the user record is checked that the user is enabled for use. It is possible to disable a user at the authorization level as well as the authentication level. This provider performs the authorization level check. If the user is disabled, the provider will issue an error to be processed according to the rules in the domain security setup (including using the Oracle WebLogic Adjudicator Providers to implement access rules).

Where would I use this Security Provider?

This provider is designed to be used in several security scenarios:

- If the domain security realm uses several authentication sources, then the Security Provider can be used to decide the order where the checks provided by this provider are executed. By default, the login even performs the same checks after all providers are called, the security provider allows that event to be done earlier in the chain.
- If the implementation is using single sign on (SSO) or federated security, this security provider can be used to decide when the checks performed are done in relation to these configurations.

By default, the checks performed by this provider are done automatically by the product login process. Use of the provider allows implementations to perform these checks earlier in the security checking process.

Implementing the Security Provider

Note: Before using the provider ensure a data source has been created to connect to the product database to access the `SC_USER` table.

Note: Each Plugin Properties must exist on a separate line

The Oracle Utilities Application Framework security provider is provided in the `$SPLEBASE/tools/bin/auth` subdirectory as `ouaf-dbmsauth-<version>.jar`. This jar file must be copied to the `$DOMAIN_HOME/lib` directory. After restarting the Administration server, the following must be configured to use this security provider:

- Login to the Oracle WebLogic Administration console using the appropriate administrator account.
- Navigate to the *Security Realms* → *myrealm* → *Providers* tab from the console.
- Select *New* to add a new Provider.

- Assign an appropriate name for the provider according to your site standards.
- Use the *CustomDBMSAuthenticator* for the Provider type.
- Use the *Ok* button to save the authenticator definition.
- Select the *Name* you assigned the provider to complete the configuration.
- Select the appropriate *Control Flag* for your site standards to determine the how the provider fits into the login sequence.
- Select the *Provider Specific* tab to configure the provider using the following settings:
    - Specify the data source created to connect to the database created earlier in the *Data Source Name* attribute.
    - Specify `com.oracle.ouaf.fed.OuafDBMSAuthenticator` for the *Plugin Class Name*.
    - Specify the `userGroup=<usergroupname>` where `<usergroupname>` is the realm group created for the product (set by `WEB_APPVIEWER_ROLE_NAME`) in the *Plugin Properties*. By default, this is set to `cisusers` if parameter not present. For example:

```
userGroup=cisusers
```

    - Optionally, specify the users you wish to bypass from this Security provider by specifying the `excludeUser=<listofusers>` where `<listofusers>` is a list of authentication users delimited by "`,`" to be excluded. For example:

```
excludeUsers=system,weblogic,OracleSystemUser
```

- *Save* the Provider configuration.
- Optionally, use *Reorder* to set the order of check.
- Optionally, configure the Adjudicator Provider for additional rules.

Chapter 16

Federated Security Support

In some security architectures, the identity used by an individual user can be shared across identity systems. Typically, this is used for cloud implementations where the product or identity is housed on a cloud system and needs to be shared across on-premise and cloud systems. This is the basis for the support of Federated Security within the product.

The Federated Security Support is supported for the following:

- Federated Single Sign On Support for the online channel.
- Support for OAuth2 tokens for Inbound Web Services.
- Support for OAuth2 tokens for inbound RESTful Web Services.
- Support for OAuth2 tokens for outbound calls to external SOAP based web services.
- Support for OAuth2 tokens for outbound calls to external RESTful based web services.

Support for OAuth2 tokens for the mobile framework.

Note: This capability supports Security Assertion Markup Language (SAML) 2.0 but is limited to authentication only. Authorization is supported in the product using the Security Model.

Note: WLST commands in this section are for illustrative purposes only and assume that the user has connected to the relevant domain using the relevant WLST commands prior to execution of the command with the relevant credentials.

Suggested References

Note: It is assumed that sites wanting to use this capability are familiar with Federated Security and the products discussed in this section. Oracle strongly recommends reading relevant documentation related to this topic prior to using this advice.

The following references are recommended to be read before proceeding using this capability:

- Oracle Identity Federation Overview

- Fusion Middleware Administrating Oracle Access Management
- Oracle Unified Directory (optional)
- Fusion Middleware Administering Security for Oracle WebLogic Server
- Using Identity Federation in Oracle Access Management
- Fusion Middleware Understanding Oracle Web Services Manager

Federated Architecture

The Federated Architecture is based upon the following components:

- An *Identity Provider* (IdP) which authenticates the SAML 2.0 based identity. This is typically an on-premise (or third party) provider that provides the ability to validate and share identity across applications/requesting systems.
- A *single sign on* product, acting as a Service Provider (SP), to detect logins and appropriately process SAML 2.0 requests and responses.
- For Web Services a SAML 2.0 security-based provider or WS-Policy compliant policy.

Prerequisites for Federated Security

Note: This support has been verified with specific Oracle technology (and selected third party providers). The instructions in this section are specific to that group of products. Alternatives may be used but instructions will need to be adjusted for those products.

The following products were used in the implementation of Federated Security:

Product	Minimum Version	Comments
Oracle Identity Management	11.1.2.3.0+	This version includes Oracle Identity Federation. If earlier version is used, then Oracle Identity Federation may need to be installed separately.
OHS/WebGate	12.2.1.4+	Includes Apache HTTP Server 2.4 which is the recommended version. The instructions in this document are based upon OHS/Webgate 11.1.2.3.0.
Oracle Utilities Application Framework	4.3.0.5.0+	
Oracle Web Services Manager	12.2.1+	Installed with Oracle Fusion Middleware Infrastructure.

Product	Minimum Version	Comments
3rd Party Identity Provider (IdP)	-	This document uses Shibboleth as an example only. Any 3rd party provider supporting SAML can be used. Refer to the Installation Reference for Oracle WebLogic and Oracle Interoperability. **Note**: This is not a recommendation from Oracle for implementing this capability. Shibboleth is recognized as a reference infrastructure.

Process Flow

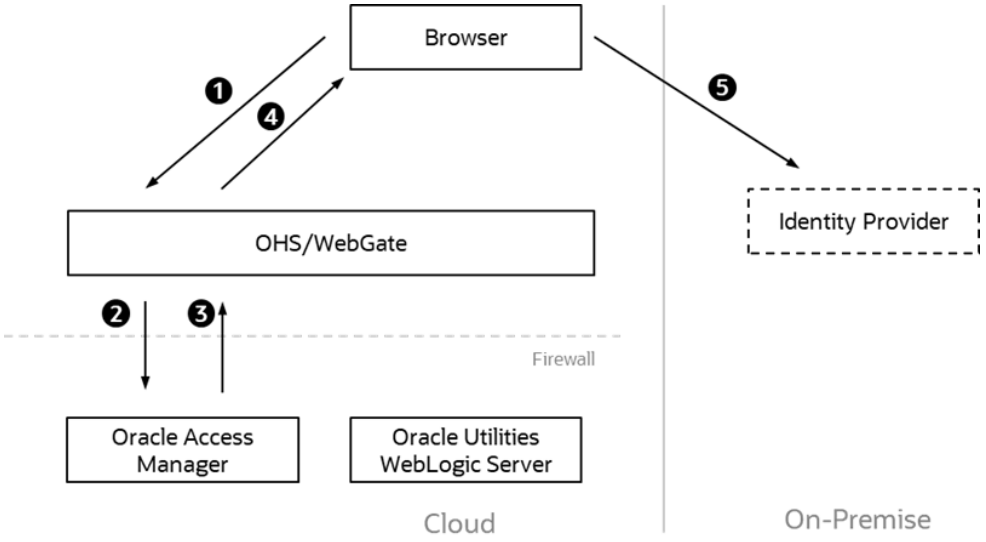
When a user logs in to an Oracle Utilities Application Framework application, there are two scenarios in relation to Single Sign On.

- **Standard Single Sign On**: This is where the single sign on infrastructure and related security repository are either all in the cloud or all on-premise. In this case the Oracle Access Management product is used as Single Sign On solution without the need for any federation.
- **Federated Single Sign On**: This is where the sign on and security provider are in a hybrid approach where one is on-premise (usually the identity provider) and one is on a cloud instance (usually the single sign on solution).

The latter has the following flow when logging in:

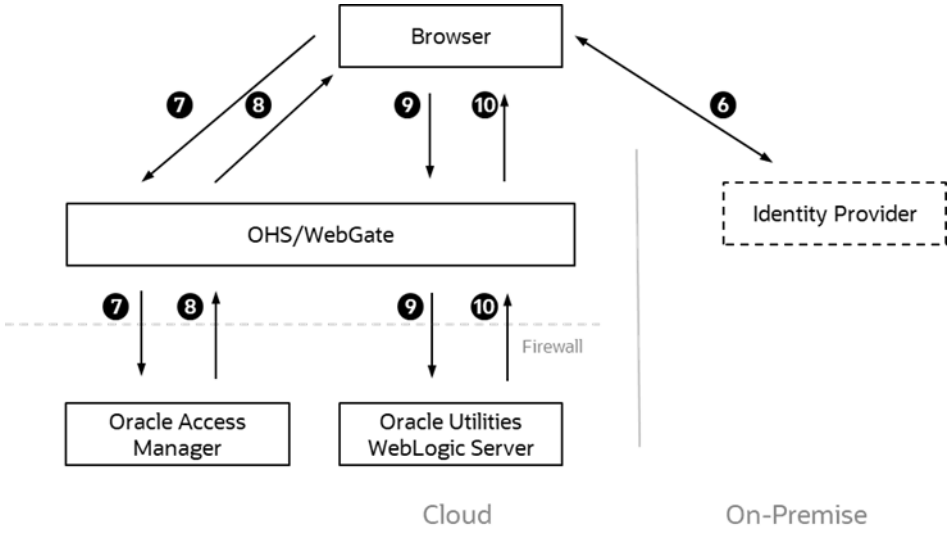
1. The user logs into the product using the provided login to the proxy (OHS/ Webgate) which houses the access rules and detects that the user has not been authenticated yet.
2. Oracle Access Manager is called to determine that validation of the authentication is the responsibility of the external Identity Provider (IdP). It formats a SAML 2.0 request to be sent to the Identity Provider.
3. It responds to the user via the product as a redirect.
4. The browser redirects the request to the Identity Provider configured by Access Manager.
5. The SAML 2.0 request is sent to the Identity Provider for processing.

This part of the flow is shown below:



6. The Identity Provider authenticates the user and responds with a SAML 2.0 assertion which includes the authentication data.
7. The browser sends the SAML 2.0 assertion to Oracle Access Manager.
8. Oracle Access Manager validates the assertion and responds with an appropriate identity assertion for the SSO session.
9. The browser sends the SSO session to Webgate which now detects the user is valid and sends all the subsequent requests in that session to the product server(s).
10. The product server(s) respond with the appropriate responses for the transactions for the duration of the session.

This part of the flow is shown below:



In summary:

- WebGate is the proxy and *gatekeeper* of the session.
- If it detects the user is valid, it will use the session information as authentication tokens for the product.

- If Webgate detects the user has not been authenticated, it will use the rules in Access Manager to determine how to authenticate the user.
- Access Manager should be configured to point to the Identity Provider and format/process any SAML 2.0 requests/responses transmitted to/from the Identity Provider. It also provides the SSO credentials for valid users.
- The Identity Provider is responsible for taking the SAML 2.0 request, validating the user in the request and sending an appropriate SAML 2.0 response.
- The browser acts as the interface, known as the *User Agent*, between the various components as it is the common point in the hybrid architecture. The feature uses standard features of HTTP to redirect traffic as needed.

Federated Online Authentication

The most common scenario is to federate the authentication of online users to a hybrid cloud solution.

Overview

The process for configuring an architecture for federation is as follows:

- Install/Provision all the software on the various platforms ready for configuration. The installation and configuration documentation will outline the steps involved in getting the software installed and configured at a basic working level.
- Identity Provider Configuration - Setup your Identity Provider (IdP) to point to your security repositories and accept the necessary SAML 2.0 request to format the appropriate SAML 2.0 response.
- Oracle HTTP Server/WebGate Configuration - Configure your proxy servers to accept requests and, optionally, support reverse proxy (to detect the client connection).
- Define Identity Provider Partner in Oracle Access Manager – Configure the access management solution with the details of the Identity Provider.
- Enable Just In Time Provisioning in Identity Federation – Whilst the user would exist in the security repository and the product, it needs to be in the access management repository on first use.
- Define WebGate Agent – In a hybrid solution agents need to be configured to enable communication across partners.
- Copy WebGate Agent Configuration to OHS/WebGate – As agents need to communicate effectively ensure that the agent configurations match.
- Define Authentication Policy for the Product Domain – The domains must be configured with the linking policies to link all the components together.
- Export the OAM SAML Metadata (optional) – Optionally, each provider may differing SAML formats. This step exports the suggested format from the access management software into the Identity Provider to ensure SAML responses are formatted correctly.

The steps outlined above are described in the next subsections.

Note: All configuration files shown in this section are examples for illustrative purposes only. Refer to documentation provided with products for details of related settings.

Identity Provider Configuration

The Identity Provider (IdP) in the solution needs to be configured to allow SAML requests and appropriate responses to be received and sent to the service providers (SP) accessing the solution. The actual steps involved will vary from provider to provider but in general the following must be performed:

- The Identity Provider (or its networking component) must be configured to allow connections from the browsers access to the provider. This can be hostnames, IP addresses or IP Address ranges allocated to your users. These addresses can be via Virtual Private networks or direct according to your networking policies. For example, in a Shibboleth installation, the `allowedRanges in the access-control.xml` configuration file needs to be set:

```
      <entry key="AccessByIPAddress">
        <bean parent="shibboleth.IPRangeAccessControl"
          p:allowedRanges="#{ {'127.0.0.1/32', '::1/128',
'<validIpRangePattern>'} }" />
      </entry>
```

- Oracle Identity Federation uses the `uid` LDAP attribute for user identification. Ensure your Identity Provider uses this identifier in any communication to the solution. For example, in a Shibboleth installation, the `Requester` should be Oracle Access Management (with Oracle Identity Federation installed) with `uid` as the `AttributeRule in the attribute-filter.xml` configuration file:

```
    <AttributeFilterPolicy id="<host>">
        <PolicyRequirementRule xsi:type="Requester"
value="http://<OAMhost>:<port>/oam/fed" />
        <AttributeRule attributeID="uid">
            <PermitValueRule xsi:type="ANY" />
        </AttributeRule>
    </AttributeFilterPolicy>
```

- Some Identity Providers require that the IdP be defined explicitly. For example, in a Shibboleth installation, the `idp.entityID` property must be set in the `idp.properties` configuration file:

```
idp.entityID= https://ouaf.oracle.com/idp/shibboleth
```

  Note: Refer to Entity Naming for a description of this setting.

- Configure your Identity Provider to connect to your LDAP security repository. Refer to the documentation with your Identity Provider for detailed information on this process. For example, in a Shibboleth installation, the `ldap.properties` file controls the interface to the LDAP.

  Note: If your implementation uses trust stores, ensure that the trust is also configured correctly for handshaking between the Identity Provider and LDAP security repository.

- Some Identity Providers need to understand the SAML request and response from/to the Oracle Access Management solution. The provider needs to understand how to download the SAML metadata to understand the interface.

For example, in a Shibboleth installation, the `metadata-providers.xml` configuration file needs to be configured:

```
<MetadataProvider id="HTTPMetadata"
          xsi:type="FileBackedHTTPMetadataProvider"
          backingFile="%{idp.home}/metadata
localCopyFrom_https_<OAMhost>_<port>.xml"
          metadataURL="https://<OAMhost>:<port>/oamfed/sp/
metadata">
    </MetadataProvider>
```

- Some Identity Providers specify their meta data for use with the Service Provider. Ensure the Identity Provider also has correct certificate usage (including X.509) for secure transmission of data. It is important that signing and encryption certificate match the certificates used in the architecture to avoid issues (such as "Signature verification failed for provider ID…" errors). Refer to Doc Id: 2032605.1 from My Oracle Support for more details. For example, in a Shibboleth installation, the X509 signing and encryption certificates in the `idp-metadata.xml` configuration file should match the certificates in `idp-signing.crt and idp-encrpytion.crt` files. In a default installation of Shibboleth the certificates are in the `credentials` subdirectory. Correct if necessary.
- This should complete the configuration of the Identity Provider.

Oracle HTTP Server/WebGate Configuration

Install and configure the Oracle HTTP Server (aka Web Tier) and the Oracle Access Manager Web Gate as per the Installation Guide.

It is recommended that reverse proxy be configured for Oracle HTTP Server to be used with your application to enable content served by different servers to appear as if coming from a single server.

To enable the reverse proxy functionality alter the Oracle HTTP Server `httpd.conf` configuration file with the following example section:

```
<VirtualHost *:<portnumber>>
   ProxyPreserveHost On
   ProxyPass "<context>" "https://<producthost>:<productport>/"
   ProxyPassReverse "<context>" "https://
<producthost>:<productport>/"
</VirtualHost>
```

Where:

<port>	Port Number allocated to proxy (default: 7777).
<producthost>	Host Name or cluster address for product.
<productport>	Port or cluster port for product.
<context>	The Product context as configured in the `WEB_CONTEXT_ROOT` setting in the `ENVIRON.INI` configuration file (For example: `/spl`).

For more information and alternatives refer to the Reverse Proxy Guide.

Define Identity Provider Partner in Oracle Access Manager

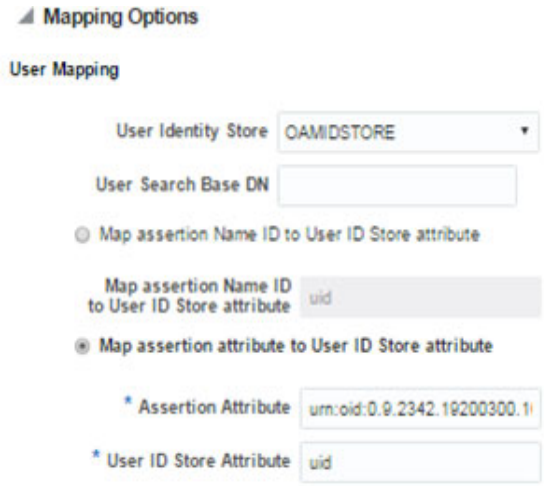
The Oracle Access Manager with Oracle Identity Federation needs to define the Identity Provider for the redirect and the integration between the products.

Oracle Identity Federation needs to be configured to identify the IdP and its interface. Refer to the Managing Identity Federation Partners documentation for details of the configuration settings.

Using the Federation → Launch Pad → Service Provider Management → Identity Provider Partner option, ensure that the following is configured:

- Enable the partner using the *Enable Partner* function in the Identity Provider Partner portal within Oracle Identity Federation/Oracle Access Manager.
- Load the SAML 2.0 Metadata that was exported from your IdP (or enter it manually). For the example Shibboleth interface, this is located in the **`idp-metadata.xml`** configuration file.
- Configure the *User Mapping* to refer to the following:
  - Select the appropriate Identity Store that was configured for your Access Management solution. This is used for Single Sign On.

The userid needs to be explicitly identified via the *Universal Resource Name* (*URN*) in the SAML metadata from the IdP metadata. For example:



  - The above attribute must be mapped to the **`uid`** attribute in Oracle Access Manager
- Optionally, set *Enable global logout* and *HTTP POST SSO Response Binding* for SSO integration.
- Within Oracle Access Manager, use the *Create Authentication Scheme and Module* option to generate and implement the definitions. If necessary, this generated configuration can be viewed or altered using *Access Manager* → *Authentication Schemes* and *Plug-ins* → *Authentication Modules* respectively.

For example, for the example Shibboleth interface refer to Oracle Interoperability for additional detailed instructions.

Enable Just In Time Provisioning in Identity Federation

The Oracle Access Manager authenticator requires that the SAML assertion from the IdP exists in the Oracle Access Manager Identity Store.

When a new user logs in to the solution for the first time, the user exists in the IdP and in the application's User object, but it also needs to exist in the Identity Store to complete the login. As the user already is in the IdP and User object, it is recommended to provision the user in the Identity Store automatically (also known as *Just In Time Provisioning*) upon first login.

This is achieved by setting the `userprovisionenabled` configuration setting to `true` within Oracle Access Manager. Refer to JIT User Provisioning in OIF / SP and Oracle Access Manager Administration for more information.

Configure the credentials for access setting the Bind DN to the connection group and the Identity Store on the *User Identity Stores* as per the Registering and Managing User Identity Stores.

Define WebGate Agent

Within Oracle Access Management, the OAM Agent for the application must be registered and configured with the following settings at a minimum:

- Set the Security to *Simple* (or *Cert* for two way certificate implementations).
- Set *Auto Create Policies* to create default access policies.
- Set the *Base URL* to the Oracle HTTP Server/Webgate used for the product.
- Set the *Access Client Password* to the password for the Webgate.
- Set the *Relative URI* to the `WEB_CONTEXT_ROOT` setting in the `ENVIRON.INI` (prefixed with **/**) in the *Protected Resource List*. For example, **/**`spl`.
- It is recommended to add the User Defined Parameter. `authorizationResultCacheTimeout` to `0` to avoid *Invalid SAML Assertion* errors.

Refer to Introduction to Agents and Registration for more information.

Copy WebGate Agent Configuration to OHS/WebGate

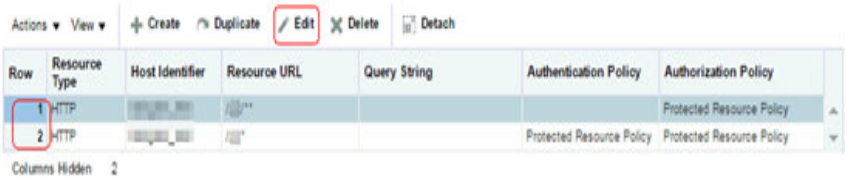
Ensure that the configuration in the previous step is available to the Oracle HTTP Server/Web Gate configuration is transferred to the `config` directory as per the Registering an OAM Agent Using the Console.

Define Authentication Policy for the Product Domain

To link the Oracle Access Manager and the IdP, an *Authentication Policy* must be configured to connect the WebGate to the IdP for communications. The following process is recommended at a minimum:

- An *Application Domain* is automatically created by the WebGate configuration implementation. This will be used by the process.
- Create an *Authentication Policy* for the selected domain with the authentication scheme appropriate for the IdP.

- Set the *Resources* to the protocol and resource URLs (set to the `WEB_CONTEXT_ROOT` setting in the `ENVIRON.INI`, prefixed with / and suffixed with *, for example /`ouaf`*) as a *Protected Resource Policy*.
- Repeat the above step for subdirectories under the context, for example, `/ouaf/**`.



- Set the *Operations* to **All** and ensure the Authentication Policy is set correctly for the IdP for each policy.

Export the OAM SAML Metadata (optional)

If the IdP requires to understand the format of the messages from the Oracle Access Manager SAML request and format the response it is recommended to export the definition using the *Export SAML 2.0 Metadata* option on the provider. Refer to Managing Settings for Identity Federation for more information.

Refer to the documentation of the IdP for how to import the settings into that product. For the example Shibboleth interface, this is not required as this is automatically performed as part of the handshaking process.

Configure the Product Identity Asserter and Authenticators

The security realm in the product Oracle WebLogic domain must be set to do the following:

- Configure the Oracle Access Manager Identity Asserter to provide the identity from the transaction flow to the product.
- Configure the Custom Authentication Service Provider that is available with the product to participate in the authentication process.
- If Oracle Unified Directory is used, as a supplemental security repository, then its Authentication Adapter must be configured to participate in the authentication process.
- If the Default Authenticator is used, as a supplemental security repository, then it must be configured to participate in the authentication process. This Authenticator is used by the administration consoles to administrate the product.

Oracle Access Manager Identity Asserter

For Oracle Access Manager to participate in the authentication the `OAMIdentityAsserter` must be added to your security realm as an Authentication Provider as per the Implementing the Security Provider. The following should be setup for federated security:

- Set the *Type* to `OAMIdentityAsserter`.

- Set the *Control Flag* to `SUFFICIENT`. This tells the security system that if the user existence is confirmed by this adapter then it is a valid user.
- Set the *Active Types* to `OAM_REMOTE_USER` and `OAM_IDENTITY_ASSERTION`. This denotes the login types used by federation.

Custom Authenticator

> **Note:** Use of the Custom Authenticator Service Provider is optional and only recommended for sites with more than one security repository that requires adjudication.

Oracle Utilities Application Framework applications ship with a Custom Authentication Service Provider, which checks that the user is defined to the product, in the User object, and that User object is *active*. Set the *Control Flag* for this Authenticator Service Provider to `SUFFICIENT`. Refer to Implementing the Security Provider for more details of this adapter.

If you are using Unified Authenticator and/or the Default Authenticator for system and administration account, then the following should be configured in the *Plugin Properties*:

- Set the `userGroup` property to the user group used for the subsetting the users accessing this plugin.
- Set the `excludeUsers` property to the comma separated list of system and administration users that are not checked by this plugin. This assumes they will be checked by other authentication providers.
- Optionally set the `debug` property to `true` for non-production debugging of your configuration to send debug log messages to the Web Server logs.

Oracle Unified Authenticator (optional)

> **Note:** Oracle Unified Directory is used in this example but any LDAP based security repository can be used in its stead using the LDAP Adapter.

If you have administration users that are not defined to your IdP but need to administrate the domain then they should either be defined in the Default Authenticator or an Oracle Unified Authentication Provider. For the latter, refer to Using OUD as a WebLogic Authentication Provider or Configuring An Authentication Provider for Oracle Unified Directory. Set the *Control Flag* for this Authenticator Service Provider to `SUFFICIENT`.

Default Authenticator

Typically, implementations would continue to use the inbuilt Default LDAP server (`DefaultAuthenticator`) for the user identities used by the product to start and stop each component (for example, the `system` or `weblogic` account). It is highly recommended that these types of accounts need to be defined in at least one security repository defined in the security realm. Set the *Control Flag* for this Authenticator Service Provider to `SUFFICIENT`.

Provider Ordering

- The most important aspect of configuring multiple authentication providers is to order them in the right order to optimize the login process. Reordering can be performed as outlined in Changing the Order of Authentication Providers with the following guidelines:
- It is highly recommended that Oracle Access Manager Identity Asserter and Custom Authenticator be placed first and second respectively. This will ensure the bulk of the users are authenticated efficiently.
- If the Oracle Unified Authenticator (optional)is used, then it can be placed in the third place to catch administration accounts. It can also be placed in second place, if it is required, but if that is before the Custom Authenticator then the group used for authentication, for example `cisusers`, must also be defined in Oracle Unified Directory.
- It is recommended that the Default Authenticator be placed last as it is typically only used for the accounts that start and stop the product.

Configure CLIENT-CERT

The last step in the online federation process is to configure the login method used by the application. For federation, it is highly recommended to set the login method to `CLIENT-CERT` which tells the product domain that authentication is coming from an external source. Failure to set this correctly will result in the login screen to be displayed upon connection, which is not desirable in a Single Sign On based solution.

Refer to the *Server Administration Guide* supplied with your product to set this value.

Federated Web Services

Product based SOAP and REST Web Services are secured and federated using Oracle Web Services Manager (OWSM) and the Oracle Mobile/Social OAuth2 service for the Mobile Framework installed as part of the Oracle Access Manager/Oracle Identity Federation installation.

> **Note:** The product supports OAuth 2.0 and uses the "2-legged" Authorization model. Refer to Understanding OAuth2 with Oracle Web Services Manager and Oracle Access Management OAuth Service for more details.

Overview

The architecture of the Web Services federation is similar in nature to the online federation with the following important differences:

- Oracle HTTP Server and WebGate are used in a similar role but delegate, via exclusion, to Oracle Web Services Manager rather than to the IdP and Oracle Access Manager.
- The architecture has an authorization server to determine the validity of the user. An authorized user is issued an access token that is used for transaction security.

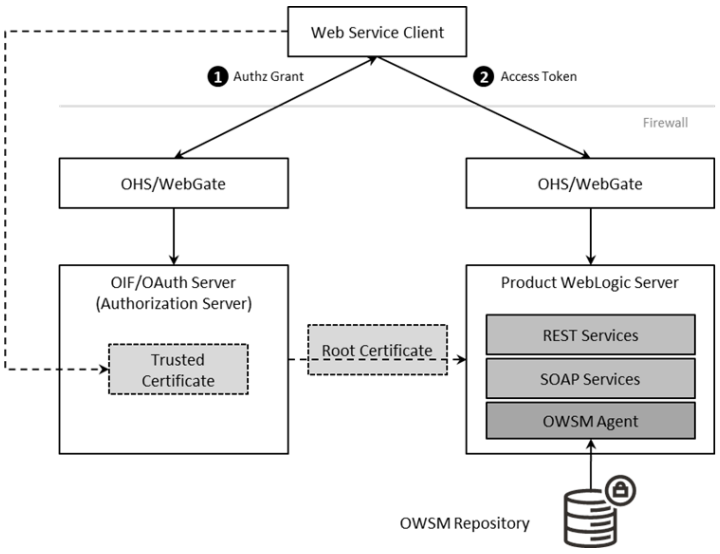
In this document, Oracle Identity Federation is used but any OAuth 2.0 compliant authorization server can be used.

- Inbound Web Services and REST use the Oracle Web Service Manager integration to implement OAuth2 with support inline and at the container level.

Process Flow

The flow of the Web Services transaction is as follows:

- The Web Service call is redirected to the Authorization Server to be granted access to the requested resources. Optionally, depending on the policy used access certificates can be used.
- If the Web Service identity is valid on the Authorization Server an Access Token is issued that is used for relevant calls for the Web Services.
- Trust is established between the Authorization Server and Product Server to ensure proper security protocols are maintained between the servers.
- For example:



Set Up OAuth Service

The first step in the configuration process is to setup the OAuth component for Oracle Identity Federation in the Authorization Server. Refer to Managing the Oracle Access Management OAuth Service for instructions on this process.

Configure WebGate for SOAP/REST communications

The online WebGate configuration needs to be altered with the following changes for securing SOAP and REST based services:

- Alter the Application domain within the Authorization Server to add the SOAP and REST URLs as separate resources.

  Where `<context>` is the value in `WEB_CONTEXT_ROOT` variable configured in the `ENVIRON.INI`.

Protocol	Resource URL
SOAP	/<context>/webservices/*
REST	/<context>/rest/ouaf/**

- Ensure the *Protection Level* is set to `Excluded` to delegate the security to Oracle Web Services Manager for these resource URL's. This will disable the *Authentication Policy* and *Authorization Policy* for the resource URL's above.

Create OAuth Client

In the online federation, an identity domain was created for use with online users. For integration with Web Services, a new identity domain must be created and configured specifically for Web Services. Refer to Configuring OAuth Services Settings for detailed information on how to create a new domain and generate a new client to interface to the domain.

When creating the new domain, the following recommendations apply to the configuration:

- Ensure the *Refresh Token Enabled* is set to `true`.
- Ensure *Lifecycle Enabled* is set to `true`.
- Ensure the *endpoints* are unique across domains. For example, `ouafoauthservice`
- Set the *Attributes* to match your site settings:

Attribute	Comments
`jwt.CryptoScheme`	To a valid scheme used in your implementation. For example, RS256.
`createdByDefault`	Set to true.
`jwt.cert.alias`	Set to oauthkey. This alias will be added later to the keystore.

- In the OAuth Web Service Client for the above domain, specify the default identity, in the *Client Id*, to be used for Web Services. This user must be defined to the product User object used for authorization. Generate the *Client Secret* for the identity. Ensure *Shown in Clear Text* is set for the identity. Under *Grant Types* under *Privileges*, ensure `Client Credentials` is enabled to make sure that, when using a JWT token, the token for the Web Services is generated correctly. Refer to Understanding OAuth2 with Oracle Web Services Manager for additional advice.

Using Keystores and Credentials

As keystores and credentials are typically used, they must match each component of the architecture so that communication can proceed. This subsection will outline the steps to generate, import and configure the various identities in keystores to provide secure communications.

Setup Oracle Web Service Manager Client

The Oracle Web Services Manager must define and generate keys to be used in the architecture:

- **Define the OWSM Credential to the Credential Store Framework (CSF)** - Add the Client identifier and the Secret generated in the OAuth Web Service Client. using the WLST command below or using Fusion Middleware Control console:

  ```
  createCred('oracle.wsm.security', key=<key>', user='<user>',
  password='<secret>', desc='<description>')
  ```

  Where:

<key>	The key used for the credential. This must match either `oauth2.client.csf.key` or the override specified in the `F1-OAUTH` Extended Lookup. For example, `f1.oauth2.client.credentials.`
<user>	The product user specified on the OAuth Web Service Client
<secret>	The value generated for secret on the OAuth Web Service Client.
<description>	A short description for the key. This can assist in finding the key in the domain.

- **Generate the Client JWT Bearer key pair** - Using `keytool` generate the `orakey` key pair and certificate into a temporary JKS keystore. For example:

  ```
  keytool -genkeypair -noprompt -keyalg RSA -keystore /tmp/orakey-
  keystore.jks -storepass '<pw>' -alias orakey -keypass '<pw>' -
  validity 7200 -dname "<cn>"
  ```

  Where:

<pw>	The password to use for the store. These values will be used in the OPSS Keystore Service (KSS) once imported.
<cn>	CN String for certificate

  Note: This example is for non'production only. Alter command for alternative certificates.

  Note: In non-production the WebLogic demo certificate can be used. It is recommended to use a valid third-party certificate be used for production. Refer to Using Third Party CA Signed Certificates to for more information.

- **Define the OPSS Keystore Service (KSS) and import keys**: Using Fusion Middleware Control or WLST create the KSS and import the `orakey` pair. For example:

  ```
  svc = getOpssService(name='KeyStoreService')
  svc.createKeyStore(appStripe='owsm', name='keystore',
  password='<ksspw>')
  svc.importKeyStore(appStripe='owsm', name='keystore',
  password='<pw>', aliases='orakey', keypasswords='<pw>',
  type='JKS', permission=false, filepath='/tmp/orakey-
  keystore.jks')
  ```

  Where:

<ksspw>	Keystore Service Password
<pw>	The password used for generated file to be imported.

- Export the certificate for use on the Authorization Server using the following WLST command. For example:

```
svc.exportKeyStoreCertificate(appStripe='owsm',
name='keystore',  password='', alias='orakey',
keypassword='<ksspw>', type='Certificate', filepath='<file>')
```

Where:

<ksspw>	Keystore Service Password
<file>	Fully qualified directory and name for exported file. For example: `/tmp/orakey.cert`

Setup Authorization Server

On the OAuth Server the following must be performed:

- **Import the Certificate**: Import the `orakey` certificate from the client into the OAuth server using the `keytool` command. For example:

```
cd $DOMAIN_HOME/config/fmwconfig
keytool -importcert -file <file> -trustcacerts -alias orakey -
keystore default-keystore.jks
```

Where:

<file>	Fully qualified directory and name for file to be imported. For example: `/tmp/orakey.cert`

- **Generate the JKS Key Pair:** Generate the JKS Key Pair to ensure that the tokens used are signed appropriately using the `keytool` utility. For example:

```
keytool -genkeypair -noprompt -keyalg RSA -sigalg SHA1withRSA -
keystore default-keystore.jks -storepass '<store_password>' -
alias oauthkey -keypass '<key_password>' -validity 3600 -dname
"<cn>"
```

Where:

<store_password>	The password for the key store.
<key_password>	The password assigned to the key.
<cn>	CN String for key.

- **Update OAuth CSF Credentials:** The CSF credentials within Oracle Web Services Manager must be updated to include the private key using the Fusion Middleware Control or a set of WLST commands. For example:

```
updateCred(map="oracle.wsm.security", key="keystore-csf-key",
user="owsm", password="<store_password>", desc="Keystore key")
updateCred(map="oracle.wsm.security", key="sign-csf-key",
user="oauthkey", password="<key_password>", desc="Signing key")
updateCred(map="oracle.wsm.security", key="enc-csf-key",
user="oauthkey", password="<key_password>", desc="Encryption
key")
```

Where:

<store_password>	The password for the key store.
<key_password>	The password assigned to the key.

- **Export Certificate for use in product:** The final step within the Authorization Server is to export the certificate for use in the product using the `keytool` utility. For example:

  ```
  keytool -exportcert -keystore default-keystore.jks -storetype
  JKS -storepass <store_password> -alias oauthkey -file <file>
  ```

  Where:

<store_password>	The password for the key store.
<file>	Fully qualified directory and name for export file.

Import Certificate into Product

On the product server the certificate and trust from the Authorization Server must be imported so that a trust is established between the servers. To achieve this the following steps are recommended to be performed on the product application server:

- Transfer the exported certificate file from the Authorization Server.
- Convert the format of the certificate file into PEM format. For example:

  ```
  openssl x509 -inform der -in <file> -out <pemfile>
  ```

  Where:

<file>	Fully qualified directory and name for exported certificate file.
<pemfile>	Fully qualified directory and name for generated PEM file.

- Import the PEM certificate into the OPSS keystore used by the product using Fusion Middleware Control or WLST commands. For example:

  ```
  svc = getOpssService(name='KeyStoreService')
  svc.importKeyStoreCertificate(appStripe='owsm',
  name='keystore', password='', alias='oauthkey', keypassword='',
  type='TrustedCertificate', filepath='<file>')
  ```

  Where:

<file>	Fully qualified directory and name for PEM file.

Enable OAuth on Product

To use OAuth within the product the following is recommended to be configured:

- Using the `configureEnv.sh -a` command set the following settings:

Attribute	Recommended Setting
CSRF Protection for REST Services	`false`
OWSM Protection for REST Services	`true`

Note: Execute the `initialSetup.sh` utility to reflect the changes.

- The following Oracle Web Service Management Policies must be attached with Oracle Web Services Manager or within the product configuration for both REST and SOAP:

Protocol	Oracle Web Services Manager Policy
SSL	`oracle/multi_token_over_ssl_rest_service_policy`
Non-SSL	`oracle/multi_token_rest_service_policy`

Use Oracle Web Service Manager Policies

The above Web Services policies must be configured within the product to configure OAuth2 support. This may be done at the individual service level (when part of your solution is federated) or globally for all services.

Individual Web Service Policies

To attach the token policies to individual Inbound Web Services the following process is recommended to be performed:

- If it does not already exist, create a Web Service Annotation of type *Annotation for OWS Security Policy*. In the `uri` *Parameter Name*, specify the appropriate policy (SSL or non-SSL outlined in the previous section.
- Attach the Web Service Annotation to the relevant services.
- Deploy/Redeploy the Inbound Web Services to implement the policy. Refer to the *Server Administration Guide* for more information.

Global Web Service Policies

Note: It is possible to override the global setting on individual Inbound Web Services by specifying additional annotations.

If security policies are to apply globally across all services, then the following process is recommended:

- Create or alter the `F1_EMAILCFG` *Feature* of Feature Type *External Messages* and specify the *Default Security Type* option with the relevant security policy.
- Deploy/Redeploy the Inbound Web Services to implement the policies across services. Refer to the *Server Administration Guide* for more information.

Federated Outbound Messages

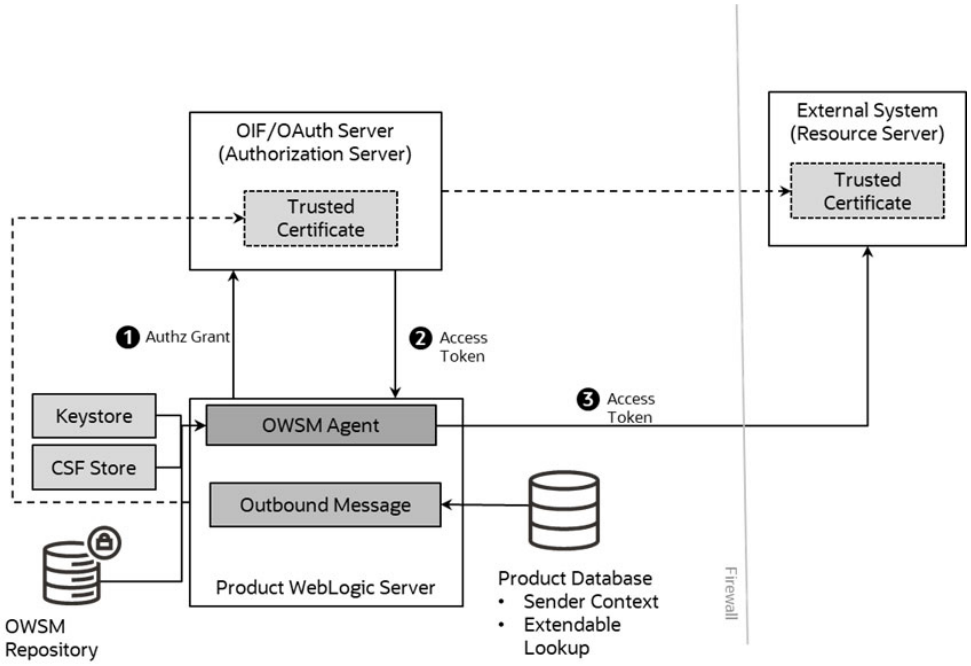
Note: The Authorization Server setup and Client Setup are identical to the configuration of Inbound Web Services. Refer to Process Flow for initial setup instructions.

Outbound Messages allows product transactions to send information out of the product via technology connectors in synchronous and asynchronous modes.

Overview

As with the Inbound Web Services the architecture of the Federated security involves issuing a token from an authorized user from the Authorization Server to the External System:

- An *Outbound Message* is created by the business process of a certain type and with the relevant payload.
- An *External System* definition with the product decides how this information is sent to the External System (the technology and mode).
- A *Message Sender* is configured to use the relevant policy in its context parameters to send to the external system. The policies supported are configured on an *Extended Lookup* to filter the policies available.
- When sending the information out, the Authorization Server issues a token to an authorized transaction to use in the transaction to the external system.
- Trust is established between the Authorization Server and Resource Server for the external system to accept valid transactions. This certificate must be exported from the Authorization Server and imported into the technology used by the Resource Server.
- For example:



OAuth Policies

To use this facility the following policies are recommended for use from Oracle Web Service Manager and the product:

- oracle/oauth2_config_client_policy
- oracle/http_oauth2_token_client_policy

These two policies are compatible with the oracle/multi_token_rest_service_policy used for Inbound Web Services (REST and SOAP).

Extendable Lookup Configuration

Note: These values are shipped with the product meta data and the policy configuration values set for the policy used should refer to the Oracle Web Services Manager documentation for a description of the valid values.

The following Extended Lookups are provided to be used:

Extended Lookup	Recommendations
F1-ValidPolicies	Two policies exist (F1-OWSM-CLIENT and F1-OWSM-TOKEN). These are delivered with the parameter settings.
F1-SetOfPolicies	This Extended Lookup is altered to set the parameter values for the valid policies above as a *Policy Set*. An extended lookup value is recommended to be added for each external system interfaced.

The following recommendations apply to the configuration of the above policies:

- For any CSF key parameters, the keys need to be added to the CSF as outlined in the Setup Oracle Web Service Manager Client.
- The URI parameters may be hardcoded or use substitution variables as outlined in the Server Administration Guide. If substitution variables are used, they should be configured in the `substitutionVariable.xml` configuration file. For example:

```
<uriVariable>
   <name>F1_TOKEN_URI</name>
   <value>http://<server>:<port>/ms_oauth/oauth2/endpoints/
oauthservice/tokens</value>
</uriVariable>
```

- Additional parameters may be set according the documentation for the Client and Token.

Message Sender Configuration

The final step in the configuration of the use of federation for *Outbound Messages* is to configure the context of the Message Sender to use the *Policy Set* that was configured in the earlier step. To use the federation the following content types must be set:

Context Type	Recommendations
Sender Security Type	This must be set to OWSM.
OWSM Policy Set	Set to the Policy Set configured (e.g. F1-OAUTH)

Configuring OAuth for the Mobile Framework

Note: This capability is only available for selected Oracle Utilities SaaS Cloud Services.

Note: For customers using the included identity solution in the Oracle Utilities SaaS Cloud Services will not need to configure this capability as

it is automatically configured as part of the provisioning process. Manual configuration is only required for externalized identity solutions.

To configure the OAuth capability for the Mobile Framework, create or update the *Master Configuration* for the *Mobile Identity Configuration*. In this configuration, the following must be configured:

Field	Description
IDCS URL	The URL of the identity solution used on the cloud instance.
IDCS Client Identifier	The mobile application identifier as configured in the identity solution used on the cloud instance.
IDCS Scope	Scope of the mobile application as configured in the identity solution used on the cloud instance.
Mobile Application URL	Full URL of mobile application that this capability is scoped to. This is only provided if the default URL is not used.

Chapter 17

Securing JNDI Access

By default, the JNDI used for the Oracle WebLogic Domain is open to read access by any valid user in the domain. This behavior may not be appropriate as dictated by the security policies at your site. It is possible to set the domain permissions, after installation, to minimize access to the JNDI.

> **Note:** The instructions in this section use the JNDI facilities described in the Access policies for JNDI resources section and Resource Types You Secure Using Roles and Policies of the Oracle WebLogic documentation. Refer to that section before configuring the security of the JNDI.

The following are included:

- Securing Product Access
- Providing Additional Access to the JNDI

Securing Product Access

The JNDI registers all the resources used in the Oracle WebLogic domain for the product. For the product to operate the following is recommended:

- Administration Users should be part of an `Admin` role. Additional roles are supplied with Oracle WebLogic.
- Product users are in group designated by the `WEB_PRINCIPAL_NAME` and `WEB_ROLE_NAME` settings in the `ENVIRON.INI`.
- Optionally, it is possible to create additional groups in your security repository to allocate specific permissions. This is outlined in Providing Additional Access to the JNDI.
- View the JNDI tree for the product servers/clusters in the console and assign the following policies:

Resource	Role/Group
Server Resources	Allocate to Admin role or individual administration accounts. This is required to start/stop and maintain the JNDI resources for the server.
JMX Resources	Allocate to Admin role or individual administration accounts. This is required to monitor the server from the console, Fusion Middleware Control and/or Oracle Enterprise Manager using the JMX interface. If the Oracle Management Pack for Oracle Utilities, this may also need to allocate to the Product group/role or individual users if the credentials used for the connection are not associated with any users in the Admin group.
JDBC Resources	Allocated to both Admin role/individual administration users and Product group/role to allow access to JDBC connection pools.
EJB Resources	Allocate to Product group/role to allow access to Business Application Server.
JMS Resources	Allocate to Product group/role to allow access for MDB, Outbound Message via JMS or JMS Real Time Adapter access.

- It is also recommended to set the `weblogic.jdbc.remoteEnabled` to `false` in the `JAVA_OPTIONS` and `WLS_JDBC_REMOTE_ENABLED` variables in the `setDomainEnv.sh` utility provided with Oracle WebLogic or by ese. For example:

  ```
  -Dweblogic.jdbc.remoteEnabled=false
  ```

- If the SSL protocol is used it is recommended to set the *RMI JDBC Setting* to `Secure` on the Product Server/Cluster Advanced Settings.
- Save the JNDI changes.

Providing Additional Access to the JNDI

If third party access is required for access to the JNDI of the product then the following is recommended:

- Set up the role or group in your security repository. In a configuration where multiple security repositories exist; this identity should only exist in one repository.
- Allocate the relevant permission as outlined in Access policies for JNDI resources section and Resource Types You Secure Using Roles and Policies of the Oracle WebLogic documentation.
- The configuration of this additional access should be appended to the existing configuration.

Chapter 18

Object Erasure Support

To support data privacy concerns, the product now supports erasure functionality for master objects. This allows for removal of Personally Identifiable Information (PII) from the product whilst respecting business rules.

> **Note:** This capability is restricted to be used with master data. Transaction data is supported for erasure using the Information Lifecycle Management (ILM).

Configuration Of Object Erasure

The Object Erasure functionality provides an ability to define the following for erasure:

- An Object Erasure Maintenance Object (`F1-OBJERSCH`) that can be used to map the reassure of the object. This maintenance object should be used as a basis for the Business Object to describe the storage of the Object Erasure information for individual objects.
- A maintenance object algorithm to *Maintain Object Erasure Schedule* to define the rules and retention for the object including any obfuscation rules.
- A generic Batch Control (`F1-OESMN`) that can be used to implement the erasure or obfuscation rules in batch.

Refer to *The Approach to Implementing Object Erasure* section of the online documentation for details of the process for configuring Object Erasure.

Chapter 19

Key Ring Support

Cryptography keys may be used to provide a signature or credentials to a request so that the system recognizes that the request comes from a trusted party. Keys may also be used to encrypt or decrypt files shared between two parties.

The Key Ring object is provided to reference the keys that are used over time for a given business use case. Only one key or key pair may be active at any given time.

The following sections include information about the functionality provided to support different key ring classes for particular use cases.

- Signature Keys
- File Signing Keys
- OAuth Keys

This chapter includes the following:

Maintaining Key Rings

The Key Ring maintenance function from the Administration menu is used to add, modify, and remove key ring definitions. To navigate to the option, select **Admin**, select **Security**, and select **Key Ring**.

Once within the function you may broadcast the key ring to maintain using the broadcast icon and use the **Edit** button to maintain the definition. You may use the **Add** function to add a new key ring entry.

When adding or maintaining a key ring the following information must be provided:

Field	Comments
Key Ring Business Object	Select the type of Key Ring to create: • OAuth Keys (F1-OAuthKeyRing) • Public Encryption Key (F1-ExtKeyRing) • RSA Signature Key Pair (K1-SignatureKey) - Used by Oracle Utilities Cloud Services only
Key Ring	Name of the key ring. Custom key rings must be prefixed with CM to reduce risk of conflicts with Oracle keys.
Description	Short description of the key ring
Detailed Description	Optional, detailed description of the key ring

Save the additions/changes for the user using the **Save** function.

Generating Keys

Once the Key Ring is defined it must have at least one activated key pair. To generate a key pair, use the **Generate Key** button.

Once generated the key ring will appear in the **Key Pairs** zone with the appropriate fingerprint. To activate the key pair, use the **Activate** button to enable the key. It is recommended to only have one pair active for each key ring at most at any time. It is possible to support multiple, but this is not good security practice. Use the **View** under the Public Key column to view and pass on the public part of the key.

> **Note**: The private key is not visible from the product in line with security standards.

Using Key Rings

Key rings can be used within numerous objects within the product. Refer to the documentation for those objects on how to connect key rings. Once connected the object will appear in the **Key Ring References** zone.

Chapter 20

Redaction Rules

Oracle Utilities Application Framework applications support configurable redaction rules that allow exports using Content Migration Assistant (CMA) and Generalized Data Export (GDE) to scramble information as necessary for privacy purposes.

This capability is not used outside of Content Migration Assistant and Generalized Data Export.

This chapter includes the following:

Setting Up Redaction Functions

Before using Redaction Rules, a set of reusable Redaction Functions that describe the technique to be used to scramble information must be configured.

To maintain Redaction Functions, use the F1-RedactionFunction Extendable Lookup to define the technique to use including:

Field	Comments
Redaction Function	Identifier for function. Must be prefixed with CM for custom function entries to avoid conflicts with base provided functions
Description	Short description for function
Override Description	Override description to allow implementations to override short description of base provided functions
Detailed Description	Detailed description of function
Status	Status of function. Valid Values: • **Active**. Function is available for use. • **Inactive**. Function is not available for use. Only Active functions are applied.

Field	Comments
Function Type	Type of function. Valid Values are: • **Date Mask** used to mask dates • **Number Mask** to mask numbers • **Regular Expression** to mask general fields • **String Mask** to mask strings
Date Mask	Mask of date in ISO format. For example: YYYY-01-01 sets all dates where this function is used to the first day of the year of the record.
Start Offset	Start position offset in field for Number Mask and String Mask
End Offset	End position offset in field for Number Mask and String Mask
Replacement Digit	Digit to use as replacement between start and end offset for Number Mask
Regular Expression	Regex expression to find values within data for Regular Expression Mask
Replacement Expression	Regex expression to replace values found in regular expression for Regular Expression Mask
Replacement Character	Character to use as replacement in between start and end offset for Number Mask
Digits Only	Replace Digits only in string. Used for String Mask only. For example, replacing digits in phone numbers.

Setting Up Redaction Rules

The Redaction Rule maintenance function from the Administration menu is used to add, modify, and remove Redaction Rule definitions. To navigate to the option, select **Admin**, select **Security**, and select **Redaction Rule**.

To maintain the Redaction Rules, specify the following:

Field	Comments
Redaction Rule	Identifier for rule. Must be prefixed with CM for custom Redaction Rules to avoid conflicts with base provided rules
Description	Short description for rule
Field Source Type	Source of field. Valid Values: • **Physical Field**. Field definition on a particular table. • **Referenced Field**. Set to apply to field across tables. • **XML Storage**. Element within CLOB.
Redaction Function	Redaction function to apply

Field	Comments
Table	Table Name for Field. Table must be defined as a Table object
Field	Field Name on Table. Field must be defined to Table on the specified Table object
Filter Expression	SQL WHERE Filter for field values to limit usage of redaction if necessary. For example: `OWNER_FLG = 'CM' AND…` The filter is limited to fields on the table. Leaving this value blank applies redaction to all values of the field.
Reference Field	Reference Field to apply redaction to.
XPath Expression	The XPath expression to determine the field in the CLOB to apply the rule to

Once Redaction Rules have been defined, they are automatically used by Content Migration Assistant (CMA) and Generalized Data Export (GDE). To bypass use of the Redaction Rules for Generalized Data Export, specify the appropriate value for the doNotApplyRedactRules parameter on the execution for the initial or ongoing extracts.