Security Guide

Oracle Real-Time Scheduler

Version 2.3.0.2 (OUAF 4.3.0.4.0)

E91568-01

February 2018

Security Guide, Oracle Real-Time Scheduler, Version 2.3.0.2 (OUAF 4.3.0.4.0)

E91568-01

Table of Contents

Preface

Welcome to Oracle Real-Time Scheduler Security Guide. This guide describes how you can configure security for Oracle Real-Time Scheduler by using the default features.

This preface contains these topics:

Audience

Oracle Real-Time Scheduler Security Guide is intended for product administrators, security administrators, application developers, and others tasked with performing the following operations securely and efficiently:

- Designing and implementing security policies to protect the data of an organization, users, and applications from accidental, inappropriate, or unauthorized actions
- Creating and enforcing policies and practices of auditing and accountability for inappropriate or unauthorized actions
- Creating, maintaining, and terminating user accounts, passwords, roles, and privileges
- Developing interfaces that provide desired services securely in a variety of computational models, leveraging product and directory services to maximize both efficiency and ease of use

To use this document, you need a basic understanding of how the product works, and basic familiarity with the security aspects of the Oracle WebLogic and Database security.

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

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

- *Oracle Real-Time Scheduler Server Administration Guide*
- *Oracle Real-Time Scheduler Batch Server Administration Guide*

- *Oracle Real-Time Scheduler DBA Guide*
- *Oracle Database Security Guide*
- *Oracle Utilities Application Framework Advanced Security (Doc Id: 1375615.1)*
- *Technical Best Practices for Oracle Utilities Application Framework Based Products (Doc Id: 560367.1)*
- *Batch Best Practices for Oracle Utilities Application Framework based products (Doc Id: 836362.1)*
- *Production Environment Configuration Guidelines (Doc Id: 1068958.1)*
- *Database Vault Integration (Doc Id: 1290700.1)*
- *Oracle Identity Management Suite Integration with Oracle Utilities Application Framework based products (Doc Id: 1375600.1)*
- *ConfigTools Best Practices (Doc Id: 1929040.1)*
- *Web Services Best Practices (Doc Id: 2214375.1)*

These documents are available from My Oracle Support and/or Oracle Delivery Cloud.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Note: Screen images in this document are for illustrative purposes only.

Note: Menu options in this document assume the use of Alphabetic sorting. If alternatives are used, then adjust the advice accordingly.

What's New in Security?

The security features and enhancements described in this section comprise the overall effort to provide superior access control, privacy, and accountability with this release of Oracle Real-Time Scheduler.

The following sections describe new security features of Oracle Real-Time Scheduler (Version 2.3.0.2) and provide pointers to additional information.

Federated Authentication Support

In some implementations it might be required to support a federated authentication configuration where an individual user's identity is stored across multiple identity management systems. This is a common scenario in cloud implementations where the identity is stored on-premise and the cloud needs to use this federation to complete authentication.

OWSM Support for REST Services

In past releases the REST capability inherited the security directly from the security realm on the Oracle WebLogic domain. In this release, it is now possible to alternatively secure REST services using the Oracle Web Services Manager to provide a wider range of security options.

Securing JNDI Access

It is now possible to specify additional levels of security on the JNDI to allow for remote access by third parties whilst retaining appropriate access for the product.

Introducing Security

One of the key aspects of the product is security which not only confirms the identity of an individual user but, once identity is confirmed, what data and what functions that user has access to within the product.

Security Features

Security is one of the key features of the product architecture protecting access to the product, its functionality and the underlying data stored and managed via the product.

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

- **Web Based Authentication** – The product provides a default method, using a traditional challenge and response mechanism, to authenticate users.
- **Support for JEE Web Application Server security** – The supported JEE Web Application Servers can integrate into a number of internal and external security stores to provide authentication services. The product can use those configurations, to liaise via the JEE Web Application Server, to authenticate users for online and Web Services based security.
- **Operating System Security** – For non-online and non-web service based channels, the product 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 the product 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 the product.
- **Native Web Services Security –** Web Services available from the product are natively available from the JEE Web Application Server. 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 the product 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 Real-Time Scheduler

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.
- **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.

Authentication

About 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 the product is delegated to the infrastructure used to run the product. This is due to a number of reasons:

- **Authentication scheme support** – The JEE Web Application Server supports a number of industry standard security repositories and authentication methods. These can be native to the JEE Web Application Server or additional products that can be are integrated.
- **Enterprise Level Identity Management** – Identity Management is typically performed at an enterprise level rather than managed at an individual product level. The product typically is not the only application used at any site and managing security across the enterprise is more efficient.

Online Authentication

The product delegates the responsibility of authentication of the online users to the JEE Web Application Server. This means that any integration that the JEE Web Application Server 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 the JEE Web Application Server itself.

Typically the JEE Web Application Server support one or more of the following:

- **Inbuilt Security** – The JEE Web Application Server typically 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** – The JEE Web Application Server can store, manage and retrieve security information directly from a database.

- **Operating System Based Security** - The JEE Web Application Server 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 your JEE Web Application Server 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 product 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 the product is housed in the JEE Web Application Server 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 the JEE Web Application Server.
- The JEE Web Application Server allows security policies and/or security access rules to be configured at an individual Web Service point of view. Any of the valid policies and security rules supported by the JEE Web Application Server can be used.
- Web Service management products such as Oracle Web Services Manager can be used to augment security for Inbound Web Services.

Authorization

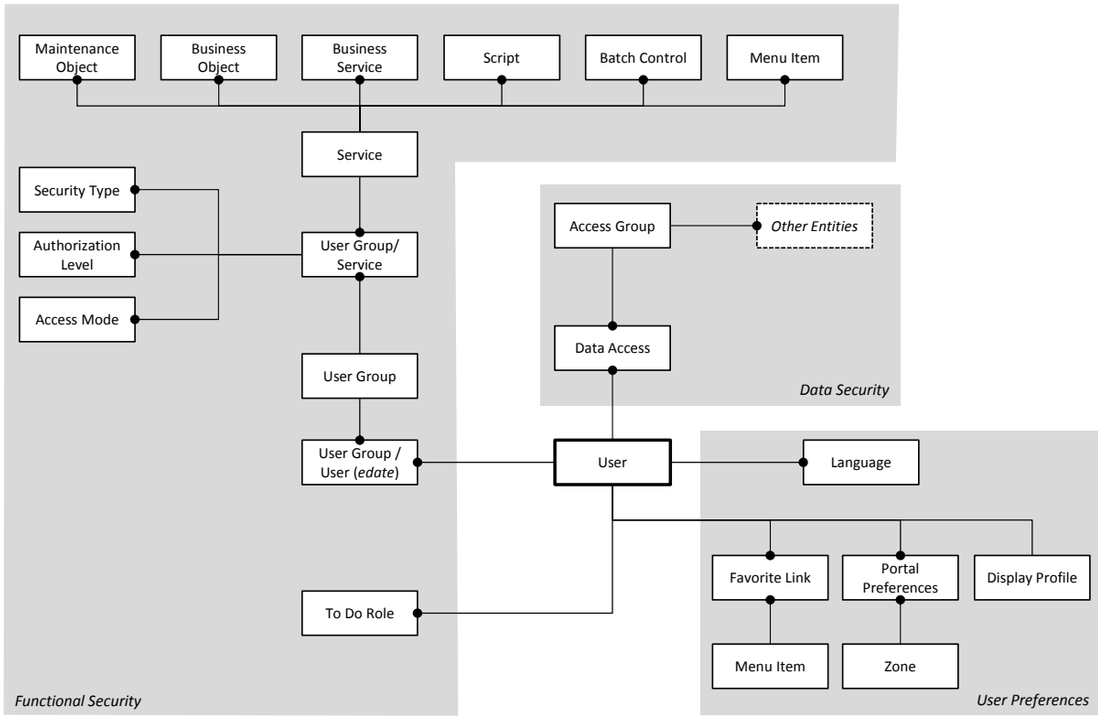
Once a user is identified they must be authorized to specific functions and data within the product.

About Authorization

The Oracle Utilities Application Framework uses an inbuilt security model for authorization. This model contains all the data necessary for the definition of authorizations to function and data. The following data model describes the security authorization model.

Authorization Model

The Oracle Utilities Application Framework uses an inbuilt security model for authorization. This model contains all the data necessary for the definition of authorizations to function and data. The following data model describes the security authorization model.



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 etc), and Language used for screens and messages and other attributes. Users are attached to To Do roles which 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 which users will process and address those errors.

Users are also attached to User Groups. This relationship is effective dated which means that the date period it is active across is also defined. This can be useful for temporary employees such as contractors or for people who change roles regularly.

User Groups are a mechanism for grouping users usually around job roles. Each User Group is then attached to the Application Services that the group is authorized to access. The Application Services are the functions within product. Loosely they correspond to each of the screens accessible in product. In this attachment the Access Mode is also defined with standards being Add, Modify, Read and Delete. With this combination it is possible to define what functions and what access is allowed to those functions for user groups (and hence users).

Additionally it is possible to define the authorization level that is allowed for the User Group to that function. 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 that 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 actually granting the user access rights to the account's bills, payment, adjustments, orders, etc.

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.

Managing Security

About Managing Security

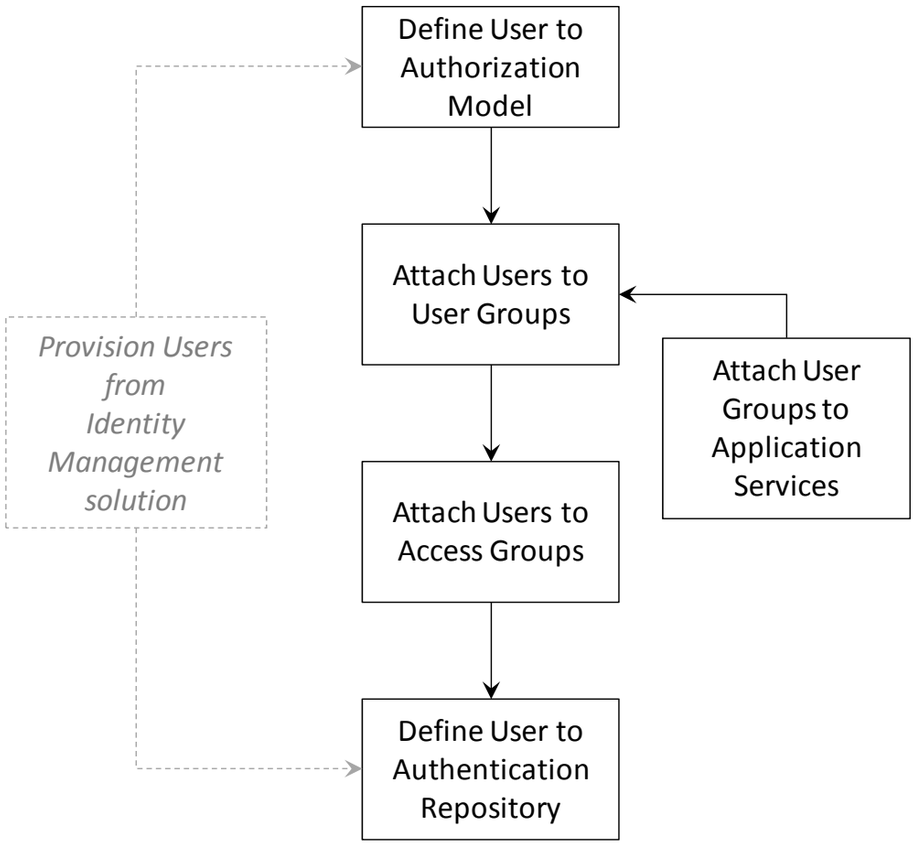
Once the security definitions are established they must be managed from the product itself, security infrastructure and security repositories used.

Managing Online Users

To manage online users a number of facilities must be configured:

- The security repository and rules must be configured in the JEE Web Application Server to enable authentication. Refer to the JEE Web Application Server *Administration Guides* for more information.
- The product group used to connect users to JEE 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 JEE 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 product. Refer to Adding Users for more details of this process.
- 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 Define 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 the product. Refer to Defining Users to User Groups for more details of this process.

Managing Users

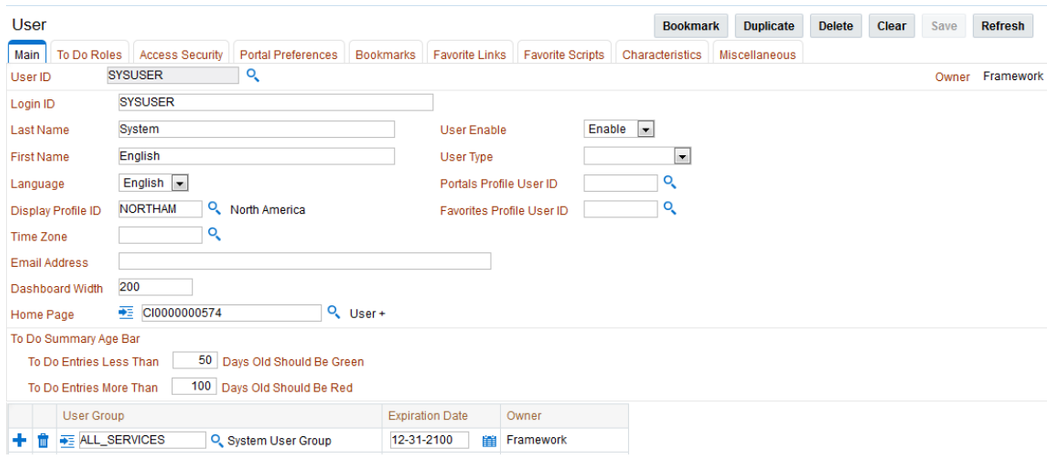
The user object in the product is used to record the security information used for identification of the user and their permissions.

The product provides a maintenance function to maintain these definitions within the product. To maintain the users the following is performed:

- Navigate to the *Administration Menu* → *UI* → *User menu* option. Using the + option on the menu allows navigation to the add function.
- The User maintenance object is displayed which maintains the security information

for a user.

- A screen similar to the one shown below is displayed:



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 User.
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 etc…

Field	Comments
Time Zone	Time Zone allocated to user account[1].
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.
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.
User Groups	This is a list of user groups and their associated expiry dates. Refer to Define 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 preferences and/or 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:

- Setup each user to be used as template and indicate the user type is set to **`Template`** to indicate such.
- For any user that will inherit the portal preferences and/or favorites specify the appropriate template user in the following fields:
  - **Portal Preferences** – Use the *Portals Profile User Id* to indicate which Template user can be used to inherit the portal preferences from.
  - **Favorites** – Use the *Favorites Profile User Id* to indicate which Template user can be used to inherit the favorites and favorite scripts from.
- Once any changes are made to the Template users portal preferences and/or favorites they will automatically apply to any attached users for these features.

[1] This feature is only applicable to specific products. Check your product online documentation for more details about applicability.

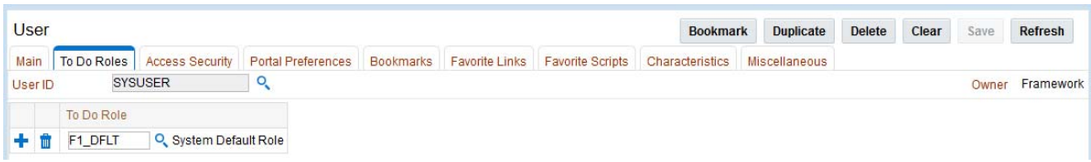
Assigning To Do Types

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 functionality. To Do roles must be setup prior to using this functionality.

The product 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 define the To Do roles for a user, navigate to the *To Do Roles* tab of user maintenance function. This will display a screen similar to the one below:



To manage the To Do Roles to be assigned to a user the following must be performed:

- Use the + to add a new To Do Role
- Use the trash icon to remove an existing To Do Role from the list.

The Search icon ( ) can be used to find the existing To Do Role or it can be typed in.

Once the users have been attached to the To Do Roles then they can access the associated TO Do types assigned to that role or any To Do directly assigned to them.

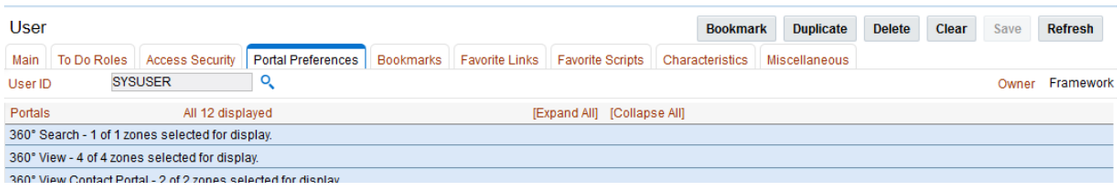
Assigning User Portal Preferences

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.

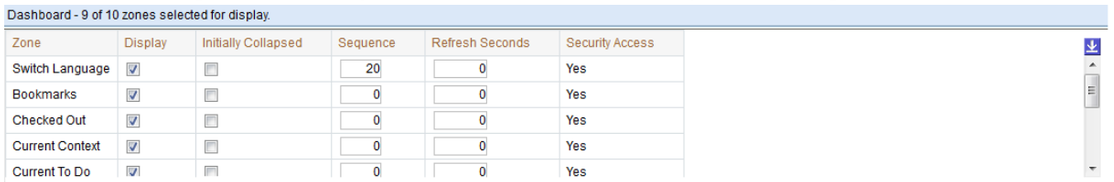
Note: Portal Preferences can be inherited from other users if Template users are used. In this case the ability to set for portal preferences for users attached to a template user are disabled.

The product user interface 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 similar to the one below 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:

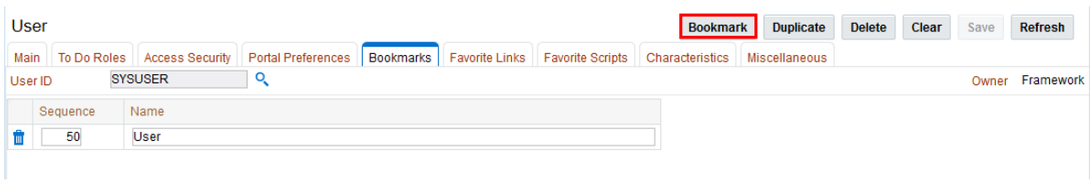
- **Display** – Whether the zone is included or not in the portal. This allows specific zones to be displayed at startup time while other zones can be hidden and only displayed upon conditions in other zones. See *Zone Visibility* in the online Administration guide for more information.
- **Initially 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 particular 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[2]. Refer to the online documentation for more information.

Assign Bookmarks

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.

Each user can attach bookmarks to their profile to access pages including the context of that page.

The definition of the bookmarks can be performed using the Bookmark button which attaches the page and context to the user profile. It is possible to view and remove bookmarks on the user profile by navigating to the *Bookmarks* tab on the User maintenance function. This will display a screen similar to the one below:



[2] While unlikely, it is possible to have a portal contain particular zones not permitted for access to an individual user.

The following fields can be set for the bookmarks:

- Sequence - Internal sequence used for sorting
- Name - Name of bookmark. The URL for the bookmark is hidden and not able to be edited manually.

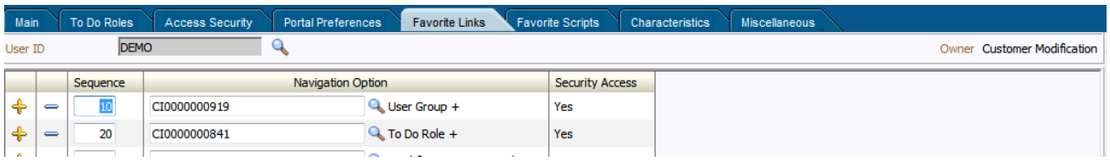
Use the to remove an existing bookmark from the list.

Assign Favorite Links

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

Each individual user can set a number of favorite functions or menu items that they can access using keyboard shortcuts or via the Favorites zone on the Dashboard.

The definition of the users Favorite Links can be configured by navigating to the *Favorite Links* tab of user maintenance function. This will display a screen similar to the one below:



The following fields can be set for the favorites:

- **Sequence** – The relative sequence number of the favorite used for sorting purposes.
- **Navigation Option** – The Navigation option to use to display the favorite. This can reference the zone or maintenance function to display when this favorite is chosen.
- **Security Access** – This is an information field that indicates whether the user has access to the Navigation Option or not.

To manage the Favorites to be assigned to a user the following must be performed:

- Use the to add a new Favorite with the appropriate Navigation Option with the appropriate Sequence to indicate where in the favorites list the option should be placed.
- Use the to remove an existing Navigation Option from the list.

The Search icon ( ) can be used to find the existing Navigation Option or it can be typed in.

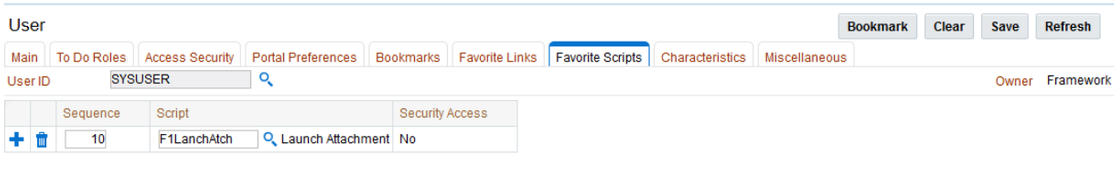
Favorites are then available to be displayed in the Favorites portal on the Dashboard.

Assign Favorite Scripts

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

Each individual user can set a number of favorite BPA Scripts that they can access using the Favorite Scripts zone on the Dashboard.

The definition of the users Favorite Scripts can be configured by navigating to the *Favorite Scripts* tab of user maintenance function. This will display a screen similar to the one below:



The following fields can be set for the favorites:

- **Sequence** – The relative sequence number of the favorite used for sorting purposes.
- **Script** – The BPA Script to use to display the favorite.
- **Security Access** – This is an information field that indicates whether the user has access to the Script or not.

To manage the Favorites to be assigned to a user the following must be performed:

- Use the ✚ to add a new Favorite indicating the Script with the appropriate Sequence to indicate where in the favorites list the option should be placed.
- Use the 🗑 to remove an existing Script from the list.

The Search icon (🔍) can be used to find the existing Script or it can be typed in.

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

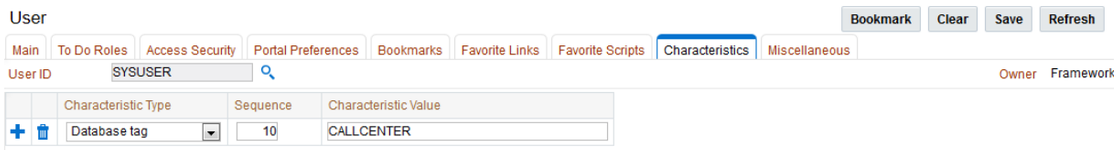
Assign User Characteristics

Note: To use this facility the appropriate characteristic types must be created and attached to the user object. Refer to the online Administration documentation for more information.

Note: The product ships with a predefined set of characteristic types.

One of the features of the product is the ability to extend the object within the product using user defined fields called *Characteristics*. Characteristics act as additional data attributes that can be used to simply provide additional information or used in custom algorithms for processing.

The user object in the product can also be customized using characteristics. This can be achieved by navigating to the *Characteristics* tab of user maintenance function. This will display a screen similar to the one below:



The following fields can be set for the favorites:

- **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** – This is the value of the characteristic. Depending on the configuration of the characteristic type this value may be free format, an attachment,

a specific format or a specific set of values.

To manage the Characteristics to be assigned to a user the following must be performed:

- Use the + to add a new Characteristic indicating the Characteristic Type, the appropriate Sequence to indicate where in the favorites list the option should be placed and the value associated with the Characteristic Type.
- Use the 🗑 to remove an existing Characteristic from the list.

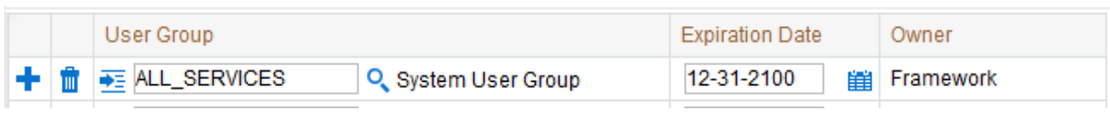
Defining Users to User Groups

To access the services within the product users must be connected to user groups which are in turn connected to application services. This defines the linkage for functionality that the user has access to.

The link between users and user groups has the following attributes:

- The linkage between users and users groups is subject to an expiry date to allow representation of transient security configurations.
- Each link between a user and user group is owned and subject to Data Ownership Rules. By default, all site created links are owned as *Customer Modifications*.
- User Groups are setup according to your site preferences. They can be job related, organization level related or a combination of factors.
- A user can be a member of any number of users groups but should be at least a member of one group to access the system.
- Users can be members of groups with overlapping permissions to application services. In the case of overlapping permissions, then the highest valid permission is used.

This can be achieved by navigating to the *Main* of user maintenance function. This will display a screen with zones at the bottom of the screen similar to the one below:



The user groups are listed that the user has access to and can be manipulated using the following:

- Use the + to add a new User Group indicating the User Group Name with the appropriate expiry date to indicate relevance of the connection. The Search icon (🔍) can be used to find the existing User Group or it can be typed in.
- Use the 🗑 to remove an existing User Group from the list.
- Use Expiration date using the 📅 to effective date the link.
- Use the context menu ⇥ to jump to the user group for more information.

The users security is then used for menu and function access regardless of access channel used (i.e. online, web service or batch).

Defining User Groups to Application Services

Note: A starter set of User Groups are loaded with the product that can be used as the basis for further security user groups.

Note: The product ships with all the application services predefined for base functions. These can be used or replaced with custom definitions as desired.

One of the fundamental security configurations for the product is to define the user groups to the application service. An application service can represent an individual service within the product, an individual menu or an individual object. When linking a user group to a service the access modes can be configured which defines the valid actions the user group can perform against the service.

Additionally each service can specify Security Types which allow for custom security rules to be applied at runtime. Refer to Security Types for more details of this facility.

To maintain the linkages between user groups and application services there are two different methods:

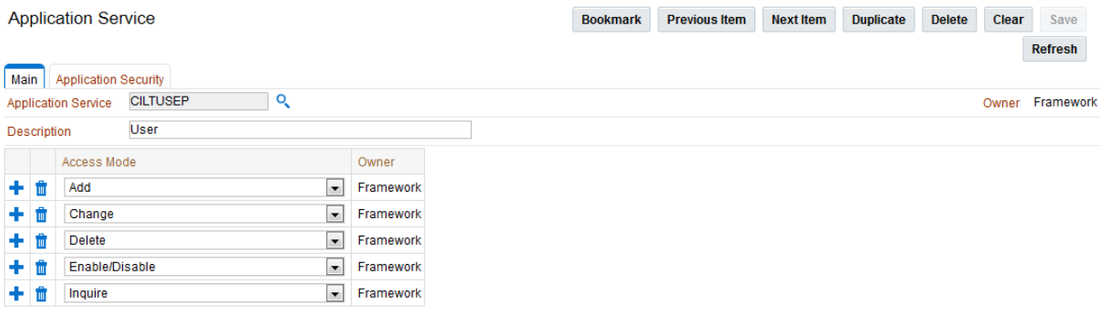
- Application Services Portal – When maintaining each Application Service it is possible to connect and disconnect the user groups and determine which groups have access to what functions.
- User Group Maintenance – When maintaining User Group definitions it is possible to connect Application Services to the group and manage users in that group from a single maintenance function.

Both methods are valid for most sites and can be used to manage the same information from different prospective.

Using the Application Services Portal

The Application Service portal allows administrators to define an application service, the valid access modes available for the Application Service and the user groups the application service is connected to.

To access the Application Services Portal, navigate to the *Administration Menu* → *A* → *Application Service* option. This will display a screen similar to the following:



On the main tab the following can be configured:

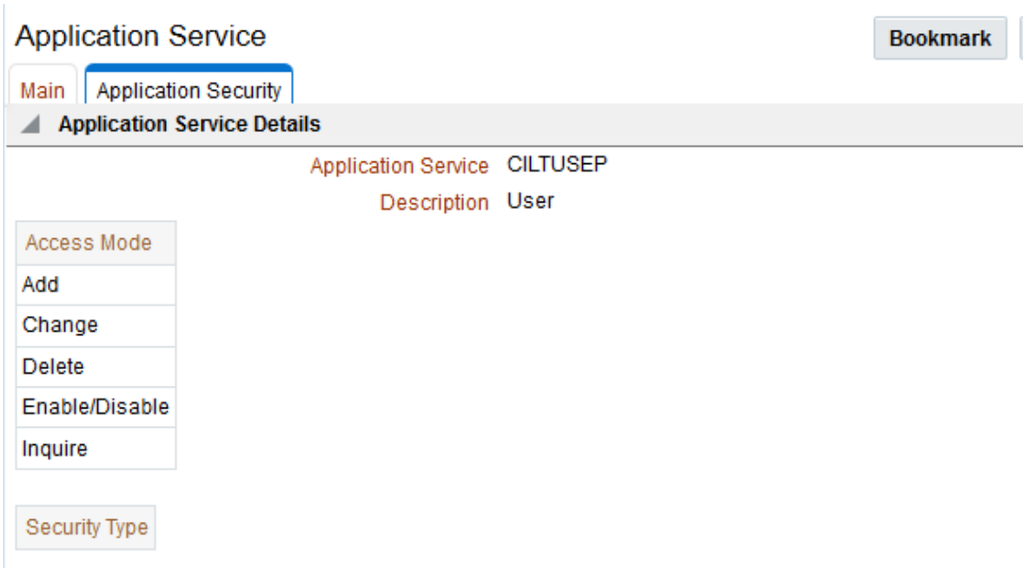
- **Application Service** – The Application Service identifier. This is a service identification token used in configuration of security on the objects, menu or service etc. For custom definitions it is recommended to prefix this value with *CM* to avoid

conflict with product provided application services.

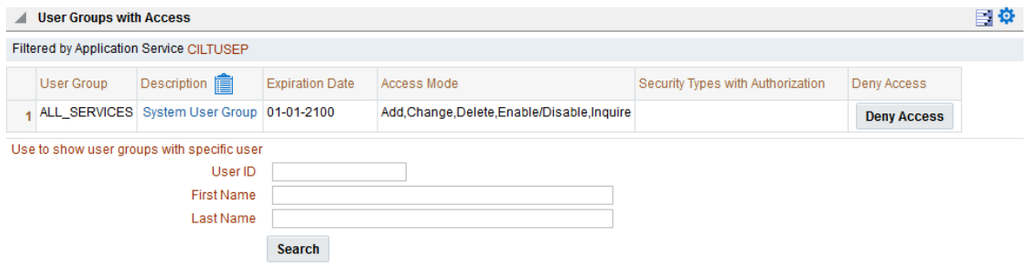
- **Description** – This is a short description used for documentation purposes. This value is displayed on any security screen when the Application Service is specified.
- **Access Modes** – A list of valid access modes is defined and displayed for the Application Service. These modes must match the internal actions supported by the objects which this Application Service is used. When using this list
  - Use the + to add a new Access mode from the drop down list of valid actions. An individual Access Mode can only be defined once for an individual Application Service.
  - Use the 🗑 to remove an existing Access Mode from the list.
  - The Access Mode link to the Application Service is ownership controlled. By default, all created links are owned as Customer Modifications. Refer to Data Ownership Rules for more details on ownership of data.

The *Application Security* tab is a portal that provides the ability to display the user group membership and manage that relationship. The portal is made up with a number of zones to provide information and maintenance capabilities:

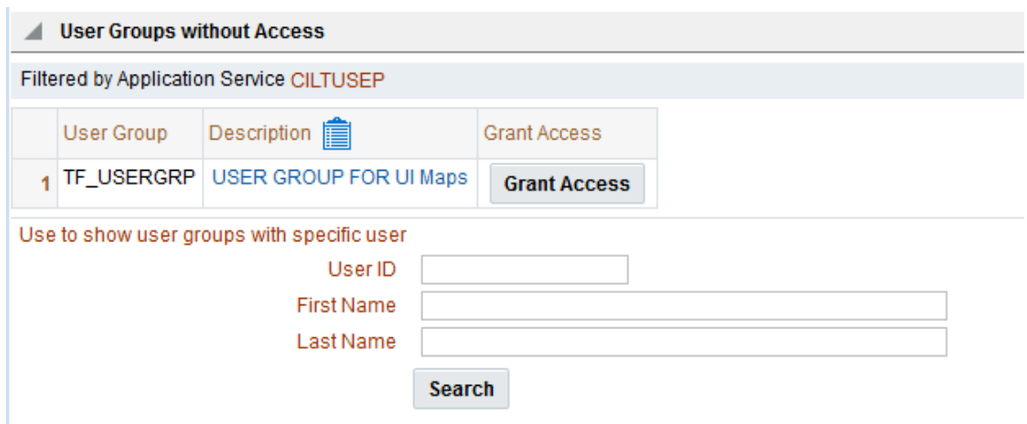
- **Application Service Details** – This zone summarizes the access modes and security types defined for an application service. For example:



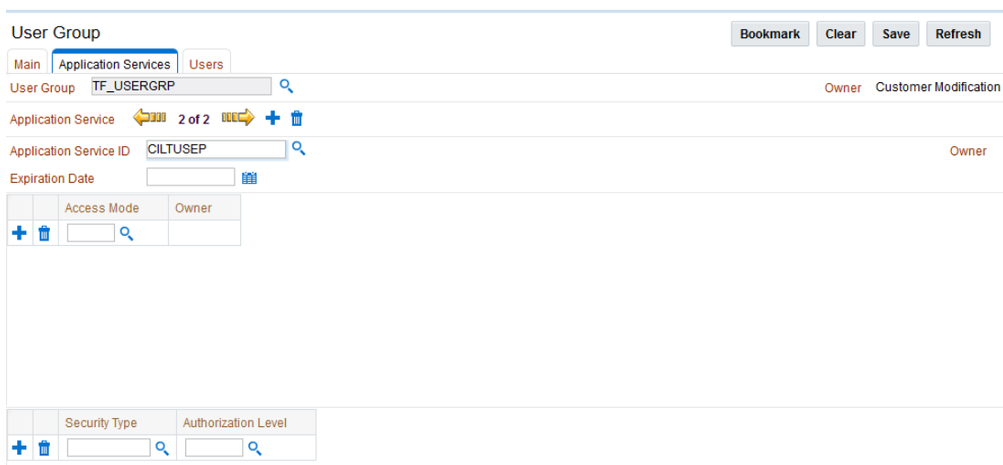
- **User Groups With Access** – This zone lists the user groups that have access to the Application Service along with the associated expiry date, access modes and security types (and associated authorization level). It is possible to deny access by a particular group to the application service using the *Deny Access* functionality. The list can be filtered to user groups for a particular user to assist in isolating particular user groups. For example:



- **User Groups Without Access** – This zone list the user groups that do not have access to the Application Service to grant access, if desired, using the *Grant Access* functionality. The list can be filtered to user groups for a particular user to assist in isolating particular user groups. For example:



Once a group is granted access then the specification of the valid access modes and security groups can be provided for the particular user group. For example:



- **Expiry Date** – Date this access will expire. Use the Date Icon ( ) to use the Date selection widget.
- **Access Mode** – Valid Access mode as defined on Application Service definition.
  - Use the + to add a new Access Mode. The Search icon ( ) can be

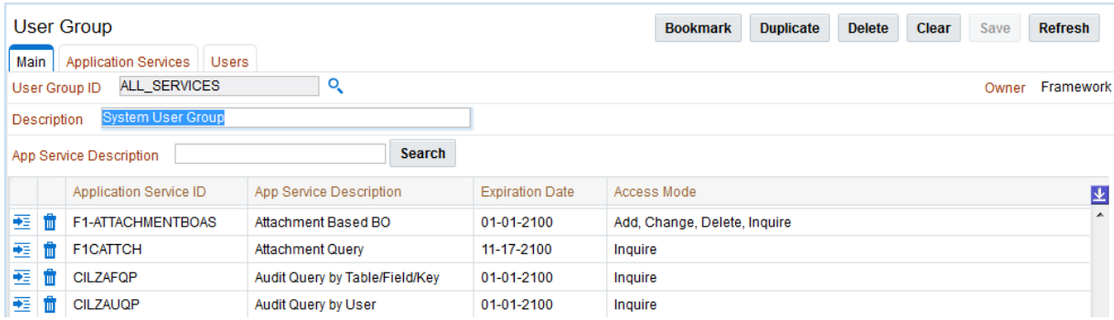
used to find the existing Access Mode or it can be typed in.

- Use the to remove an existing Access Mode from the list.

- **Owner –** Ownership of link (refer to Data Ownership Rules)
- **Security Type** – Security Type code associated with Application Service.
  - Use the to add a new Security Type. The Search icon ( ) can be used to find the existing Security Type or it can be typed in.
  - Use the to remove an existing Security Type from the list.
- **Authorization Level** – The Authorization Level assigned to this User Group when executing this Application Service for the Security Type. The Search icon ( ) can be used to find the existing Authorization Level or it can be typed in.

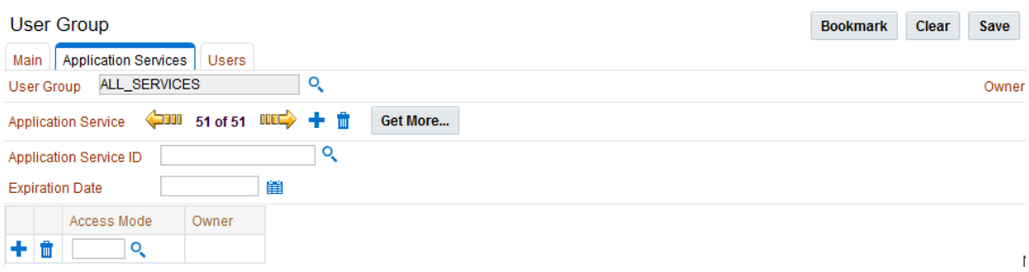
Using User Group Maintenance

When editing an individual user group it is possible to define the accessible application services and connect users to the user group from the user group maintenance function. To do this, navigate to the *Administration Menu* → *U* → *User Group* menu option. This will display a screen similar to the following:

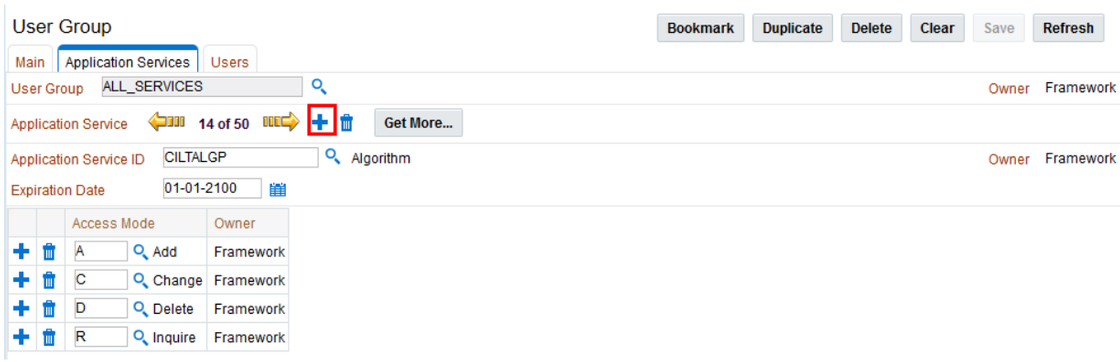


The services that this user group has access to are shown with the associated expiry date and access modes for the user group. The following actions maintain the information:

- Use the icon to edit an existing permission.
- Use the icon to remove an associate between a user group and an application service.
- Use the on the *Application Services* tab to add a new association between an application service and an individual user group. For example:



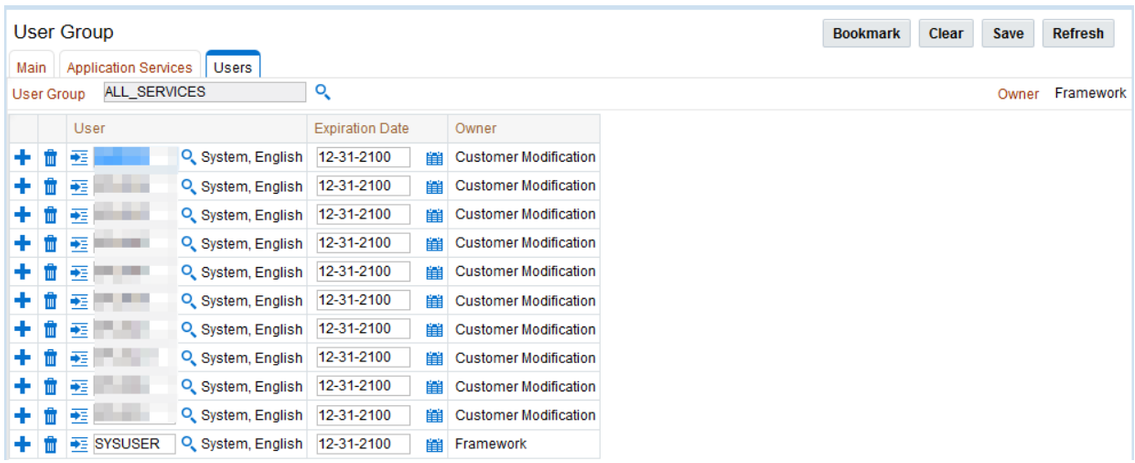
When editing an existing association or adding a new association the Application Services tab is displayed to maintain the association with associated Access Modes and Security Types. For example:



As with the Application Service Portal, it is possible to define the following from this screen:

- **Expiry Date** – Date this access will expire. Use the Date Icon ( ) to use the Date selection widget.
- **Access Mode** – Valid Access mode as defined on Application Service definition.
  - Use the to add a new Access Mode. The Search icon ( ) can be used to find the existing Access Mode or it can be typed in.
  - Use the to remove an existing Access Mode from the list.
- **Owner –** Ownership of link (refer to Data Ownership Rules)
- **Security Type** – Security Type code associated with Application Service.
  - Use the to add a new Security Type. The Search icon ( ) can be used to find the existing Security Type or it can be typed in.
  - Use the to remove an existing Security Type from the list.
- **Authorization Level** – The Authorization Level assigned to this User Group when executing this Application Service for the Security Type. The Search icon ( ) can be used to find the existing Authorization Level or it can be typed in.

Additional from the User Group maintenance screen it is possible to manage the users associated with this user group. The *Users* tab is used to define this information. For example:



The screen allows users to be associated with the user group with the following information:

- **User** – This is the authorization user identifier to be connected to the user group. The Search icon ( ) can be used to find the existing User or they can be typed in
- **Expiration Date** – Date the association between the user and user group will expire. Use the Date Icon ( ) to use the Date selection widget.
- **Owner** - Ownership of link (refer to Data Ownership Rules).

Use the to add a new User or use the to remove an individual user from the list.

Define Users to Data Access Groups

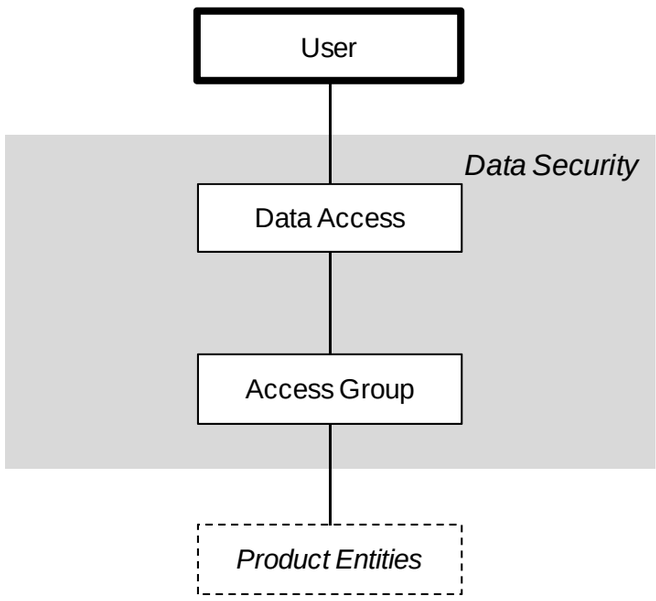
Note: Not all products support Data Access Roles and Data Access Groups. Refer to the online Administration Guide for more details.

Data Access Groups are used to define the subset of data objects the user is permitted to access. There are two levels to the definition of data access:

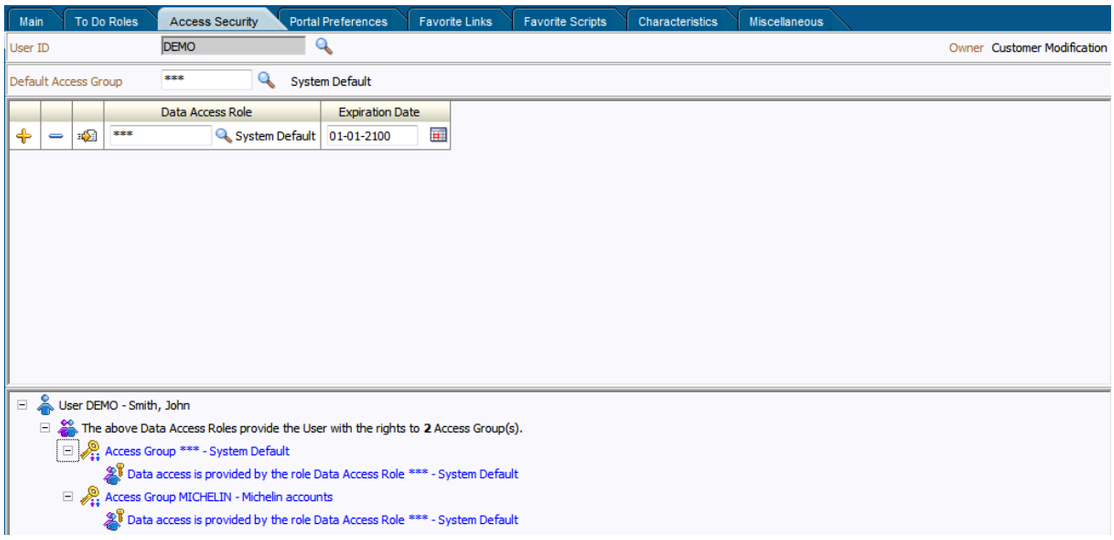
- **Data Access Roles** – User are connected to Data Access Roles which defines the groups of data permissions the user has access to. Data Access Roles are connected to Data Access Groups (a.k.a. Access Groups).
- **Data Access Groups** – Data Access Groups are tags that are attached to entities in the product to implement data security[3]. Data Access Groups are maintained using Access Group maintenance. Refer to the online Administration Guide for more details of this facility.

The relationships between these objects are illustrated in the figure below:

[3] Attaching a Data Access Group to a product entity it does not automatically implement data security. Queries for that object must be altered to take into account the Data Access Group. Refer to the Oracle Utilities SDK for more details.



To maintain the Data Access Roles and Access Groups a user has access to, navigate to the *Access Security* tab of the user maintenance function. This will display a screen similar to the following:



The screen will allow the definition and display of the following information:

- **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. The Search icon ( ) can be used to find the existing Data Access Role or they can be typed in
- **Expiration Date** - Date the association between the user and data access role will expire. Use the Date Icon ( ) to use the Date selection widget.

Use the to add a new Data Access Role or use the to remove a Data Access Role from the list.

User Enable and Disable

One feature of security is that the user record is attached to some objects for audit purposes (some objects are automatic, such as financials, and some are configurable). When a user does any work in the product and the user has been attached to some audit object across the whole product, the user cannot be deleted. This is due to auditing requirements.

There is a feature on the user object to enable or disable a user by setting the appropriate value for **User Enable** on the User object. This has the following implications:

User Enable	Implications
Enable	• User can access system. • User can process records according to the authentication model. • User must be active in Security repository to fully access the product.
Disable	• User cannot access the system regardless of other security setup • User record is retained for audit purposes only. • User does not have to exist in the Security Repository.

This facility has a number of key use cases:

- **Support for personnel (permanent or temporary) leaving** – It is possible to manually disable users once they leave the organization yet keep the
- **Logical deletion** – If the user needs to be deleted for any reason then setting Disable
- **Temporary disablement** – If business rules need to isolate users then setting the **User Enable** for appropriate users can effectively disable them from the product.

Note: When a user is disabled, it will apply when the user next attempts to login or when the security cache is refreshed.

Managing Batch Users

Each time a batch process is executed the security components of the product 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 a number of security mechanisms:

- **Authorization** – Any batch users must be defined to the operating system configured security repository and be a member of the operating system group assigned to the product.

  Note: The userid used does not have to match the authorization user used within the product.

- **Authorization** – The authorization user is defined within the product 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 Batch Administration Guide for details of the parameters used for batch processing.

To manage batch user therefore the following is recommended:

- Add the authentication user used to initiate the threadpool 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

Note: Native Security support is only supported for XAI Inbound Services using the ***BusinessAdapter*** *Adapter.*

From a product perspective a Web Service is a channel into the objects within the product. 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:

- **Authentication** – The Web Services component of the product uses the Web Services support native to the JEE Web Application Server. This allows security tokens supporting many standards to be used to authentication individual web service calls.
- **Authorization** – The Web Services component uses the same authorization model as the online user and batch components use.

  Note: The user for authentication is used to map to the authorization user within the user object 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 the JEE Web Application Server. This should match the Login Id used for the authorization model.
- Security Policies need to be attached to Web Services using the JEE Web Application Server. For Oracle WebLogic the security policies available using Oracle Web Services Manager is available for use with individual Web Services. Multiple policies are supported. Refer to the *Oracle Fusion Middleware Security and Administrator's Guide for Web Services* 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, Business 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 Web Services, refer to the *XAI Best Practices* (Doc Id: 942074.1).

Authentication User

In the user object there are two different user identifiers namely Userid and Login Id. The different identifiers have differing roles:

- The Userid, which is up to 8 characters in length, is used internally for authorization and is passed to the database connection as the **CLIENT_IDENTIFIER** on the database connection. This user cannot be changed after the user has created any

records in the system as it is used for record ownership in some objects and in auditing.

- The Login Id, which is up to 256 characters in length, is used for authentication to the security repository configured on JEE Web Application container. The Login Id can match the Userid but can differ to reflect site standards. Unlike the Userid the Login Id can changed at anytime to reflect changes in the organization such name changes or acquisition.

Note: The Login Id must match, in the same case, as the entry in the configured security repository for the JEE Web Application Server.

When maintaining a user, it is important that the Login Id is only changed using the maintenance function, LDAP Import or any XAI/Inbound Web Service based upon the USER object and not directly using other means (such as direct SQL) as 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 deemed not authorized and will be refused access to the product. To ensure security hashes are correct use the Synchronize Data Encryption function to reset the user security hash.

Advanced Security

About Advanced Security

While the default security settings are adequate for most sites, there are a number of 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.

JEE Authentication Group

The default installation of the product includes a default authentication group (**role-name**) defined within the JEE Web Application web descriptor (web.xml). This role name is used by the JEE Web Application to link the authorized users within the product to the associated JEE physical resources (i.e. pages, configuration files) within the JEE Web Application Server. The specification of the group in the web descriptor is in the security section[4]. 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 the JEE Web Application Server, this group is instantiated ready to be allocated to individual users. Users of the product must be attached to this group to use the product.

From a configuration point of view there are a number of 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 the JEE Web Application Server 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 JEE administration group is changed after installation time, users will need to be migrated to the new JEE administrations group either manually, using tools provided with the security repository or JEE Web Application Server.

[4] The security role is used in a number of sections of the web application descriptor.

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 the JEE Web Application Server, 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. The product provides a prebuilt logon screen but can be replaced with custom logon screens[5] 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

- **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.
- **CLIENT-CERT**[6] - This is an advanced configuration to allow for certificated (one way or two way) to be used. Refer to the documentation supplied with the JEE Web Application Server for more details of 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 is used by the product to determine the maintenance owner of key data as well as protect important data shipped with the product from accidental deletion.

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 flag has a number of valid values:

- **Base** – This is important information shipped with the product and cannot be deleted or modified using the delete or medication functions, respectively, regardless of user permissions. This is reserved for use by the product to ship key important information and to protect that information. Deletion of this information directly

[5] Custom logon screens should be placed in the **cm** directory of the web application server as outlined in the Oracle Utilities SDK.

[6] **CLIENT-CERT** is supported but requires manual changes to configuration files. Refer to the Server Administration Guide on implementing custom templates.

from a product database will result in unexpected results.

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

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 to the product is based upon Java Management Extensions (JMX) allowing components of the product to be managed and monitored from JSR160 compliant consoles including jconsole or Oracle Enterprise Manager.

Refer to the *Server Administration* and *Batch Server Administration Guides* 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 or **`%SPLEBASE%\scripts`** in Windows):

- **`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 and **`readwrite`** – Update access and can access update operations

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

Field	Comments
Userid	Authentication user to access JMX

Field	Comments
Password	Password in plain text or encrypted format.

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

SSL based Security

Note: For a full description of SSL setup refer to the ***To Setup SSL*** *section of Monitoring and Management Using JMX Technology.*

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

- Security has to be setup using the Simple File Based Security or Using 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 JEE Web Application Server 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: Additional options are also supported as documented in 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 Oracle WebLogic to use SSL and altering the startup scripts for Oracle WebLogic to include the above options.

Using 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 JAAS configuration stored in the **`java.login.config`** file **`$SPLEBASE/splapp/config`** directory (or **`%SPLEBASE%\splapp\config`** directory on Windows).

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`** for information and examples of login configurations.

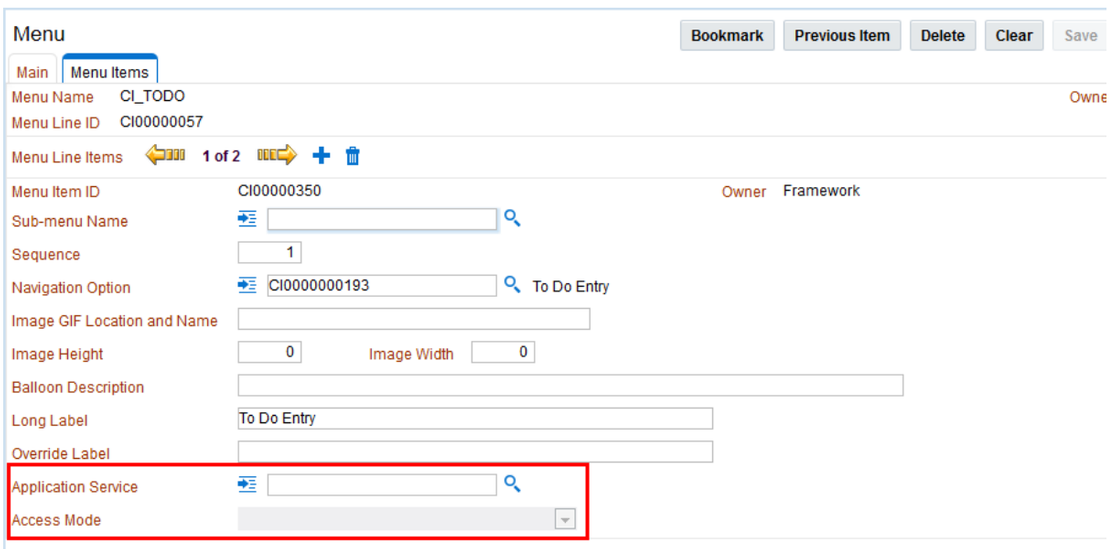
*Note: To implement the custom security source custom templates for **`java.login.config`** must be implemented according to the process outlined in the Server Administration Guide. This configuration affects all modes of access (i.e. online, Web Services and Batch).*

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 sufficient 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.

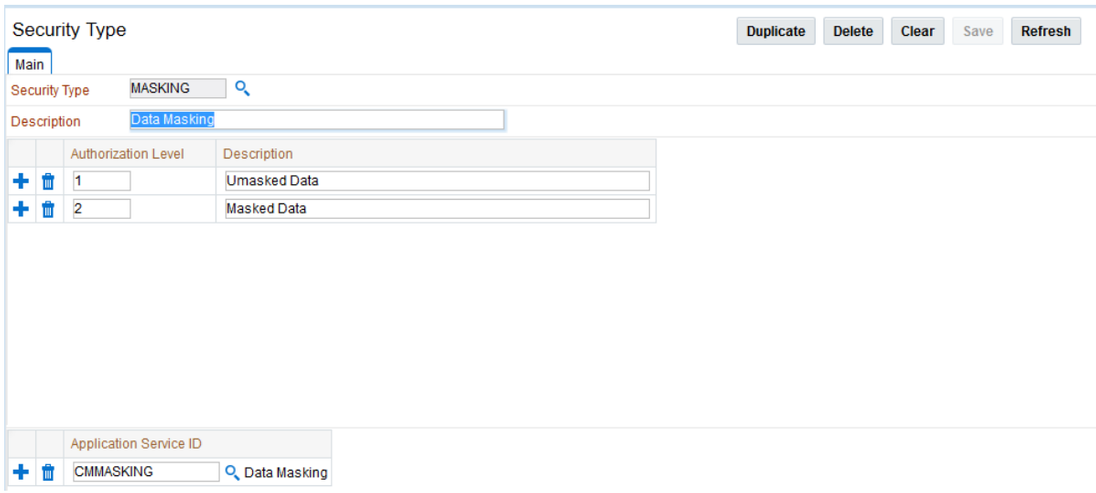
It is possible to specify the Application Service on a menu item on the *Menu Items* tab of the *Administration → M → Menu* option. For example:



Security Types

By default users have full access to the objects via the access methods specified in their user groups. If the implementation wishes 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, *Administration → S → Security Types* option to display the Security Types maintenance function. For example:



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

- **Security Type** – Identifier for Security Code
- **Description** – A short description of the use of the Security Code.
- **Authorization Levels** – A list of codes (Authorization Level) and associated descriptions. Use the + to add a new Authorization Level or use the 🗑 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 + to add a new Application Service or use the 🗑 to remove an existing Application Service from the list. The Search icon (🔍) can be used to find the existing Application Server or it can be typed in.

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

Default Generic Application Services

By default all a 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 the product 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:

- **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.

Use of these generic Application Services is optional.

Administration Delegation

By default, the product 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 sufficient 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 embedded mode and using the **spl** utility to manage the startup and shutdown of the product then the utility permissions must be altered to set the *sticky* bit[7], using the **chmod +t** or **chmod +s** command, so that the utility must run as the product administration account.

  Note: Permissions on the directories are set to restrict the administration functions. Do not alter the permissions on individual directories and file unless otherwise directed.

- 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.

[7] Support for sticky bit varies from operating system to operating system

Secure Communications (SSL)

By default, the product uses HTTP for communication 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:

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

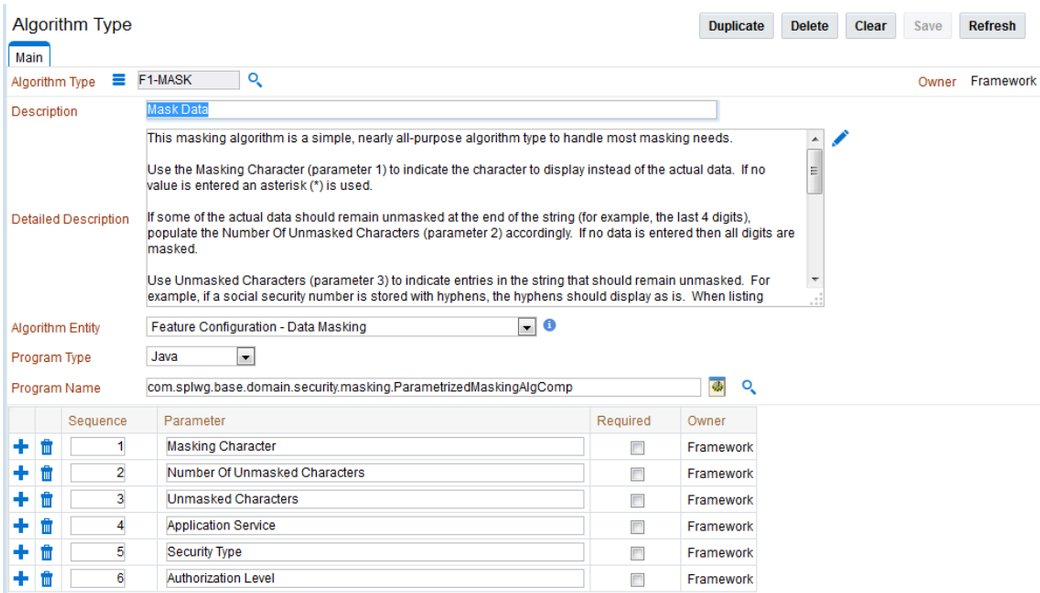
Data Masking Support

By default, the data in the product is unmasked for authorized users. If particular 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 masked fashion; it is configured to be displayed in masked format for particular users using the Security Types.

To mask data using the internal data masking capability:

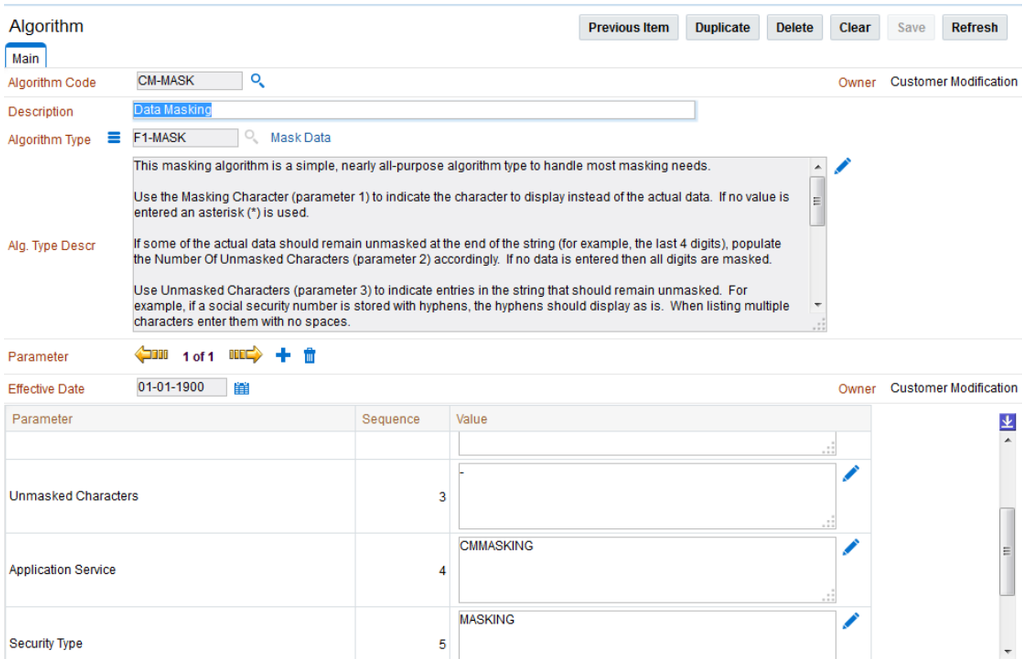
- An internal algorithm type of **F1-MASK** is supplied with the product to perform basic data masking. For example:



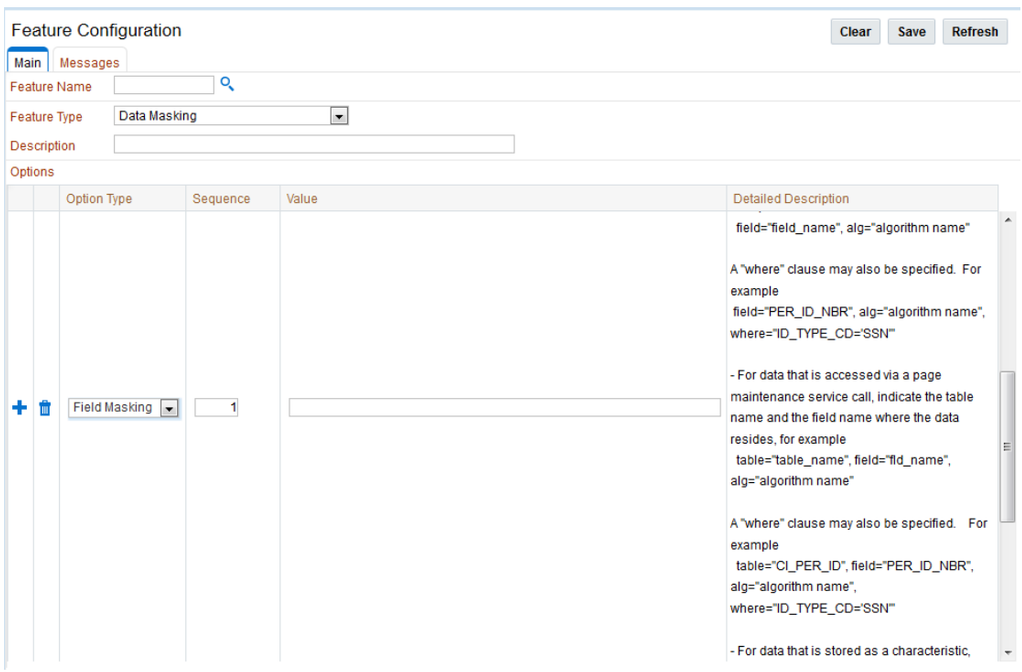
- The following parameters are applicable to the algorithm:

- **Masking Character** – The character to be used as a mask. By default the ***** character is used.
- **Number of Unmasked Characters** – 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** – A list of characters, with no spaces, to leave unmasked. Commonly, this is used to denote delimiter characters to enhance recognition.
- **Application Service** – Application Service used for security authorization checking. This allows global (or local) services to be configured to indicate security access to data masks.
- **Security Type** – The 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, Security Type and given the Authorization Level to determine the level of data masking.
- **Authorization Level** – The authorization level used to determine if a user has access to the data unmasked. 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 Administration → A → Algorithm* option. For example:



- Attach user groups to the Application Service with the appropriate Authorization Level for the Security Type.
- Create or update a Feature Configuration of Feature Type **`Data Masking`** using the *Administration → F → Feature Configuration* option. For example:



- For each field to mask add an entry to the Options section of the Feature Configuration with the following values:
  - **Option Type** – Select **`Field Masking`** for Data Masking.
  - **Sequence** – Specify a sequence number for order of evaluation.
  - **Value** – Specify the tag string, delimited by "**`,`**", to indicate the definition of the data masking with the following tags depending on how the data is accessed:
    - Only fields defined as strings are supported by the supplied algorithm.
    - To reference the masking algorithm, enter **`alg="`*`algorithm name`*`"`**. The algorithm's algorithm type must reference the Data Masking algorithm entity.
    - For data that is accessed via a schema-based object call, the field to be masked must reference a meta-data field name in its schema definition. For example, if you want to mask a credit card number, let's assume that field is defined in the schema is **`<creditCard mdField="CCNBR" mapField="EXT_ACCT_ID"/>`**. In this case, the option value should be **`field="CCNBR", alg="`*`algorithm name`*`"`**.
    - For data that is accessed via 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. This 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 that is stored as a characteristic, simply indicate the characteristic type **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 on

Note: Only ad-hoc characteristics are supported at the present time.

- For data that is 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 name of the search service, launch the search in question, right click in the filter area and choose *View Source*. Search for *ServiceName*. The service name is listed there. To find the field name to mask, go back to the search window and right click on the results area and choose View Source. Look for the Widget Info section and find the field name in the SEARCH RESULTS (do not include the $).

*Note: The **WHERE** statement can only apply to fields that are also part of the search results.*

- Use the  to add a new Data Masking definition or use the  to remove an existing Data Masking definition from the list.

Securing Files

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

The product file structure is protected by permissions set at the operating system level. By default, the settings provided with the product 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 product *Server Administration Guide*.

If at any time, the permission are manually altered and need to be reset to the defaults then the following process can be used:

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

The environment permissions will be reset to the defaults supplied with the product.

Password Management

On a regular basis passwords are changed to maintain security rules. The product uses a number of passwords that may require changing on a regular basis. The following table lists all the passwords used in the product and guidelines for changing the password values used by the product.

Password Owner	Location	Comments
Online User	JEE Authentication Source	No configuration changes. User changes password in security repository directly or indirectly using security products. Security repository is configured in JEE Web Application Server[8].
Web Service User	JEE Authentication Source	No configuration changes. User changes password in security repository directly or indirectly using security products. Security repository is configured in JEE Web Application Server.
Batch User	Operating System	No configuration changes. User changes password in security repository directly or indirectly using security products. Security repository is configured in operating system.
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.
JEE 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

The product features an online debug mode which is used for problem solving and development personnel to trace their code or 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

[8] **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.

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 product features an online cache management facility which is used to reset the online cache to force new values to be loaded. As with other functions within the product the cache management function is security controlled.

To use this facility any of the user groups an individual user must include *Change* access to the **F1ADMIN** application service. This will enable 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 product. The security settings for the Inbound Web Services can be summarized as follows:

- Inbound Web Services rely on Web Services standards supported by the JEE Web Application Server for authentication support.
- Inbound Web Services supports the WS-Policy standards supported by the JEE Web Application Server 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
    - Element Level policies are not supported at the present time.
    - 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 a number of 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 used to translate calls from authentication users not known to the system.
ouaf.ws.deploy.user	Administration user used 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) header 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:

- SOAP Insert Time Stamp - A set of timestamps can be added to the transaction to support WS-Security to avoid replay attacks.
- Additional SOAP Security inbuilt - The SOAP Header can now has 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.

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

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

Groovy Support

The product 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 blacklisted for security reasons and therefore cannot be used. Alternative functions are provided to provide safe access to selected operating system functions.
- It is possible to implement a custom whitelist for non-cloud implementations. Refer t 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, the use of **`FILE-PATH`** batch variable was restricted to local mounted storage[9]. 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 the following is recommended:

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

Connection Details	Comments
File Adapter	Use *Oracle Cloud Object Storage*.
REST Endpoint URL	Endpoint URL for cloud storage. Exclude the Service Name/Container Name from the URL
User Name	Cloud User Name to use
Password	Password for Cloud User Name

- 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>/<serviceName>`
  Where:

  `<ExtendableLookupValue>` Name of lookup value configure in Extendable Lookup

[9] It was possible to use network storage via mapped directories.

	F1-FileStorage
<serviceName>	Service Name for Oracle Cloud Object Storage Service

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 the product 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.

The **SYSUSER** account cannot be physically removed as it is used by the initial installation, and is owned by the product, 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 that is used by the implementation, that uses **SYSUSER** as the default identity, has been changed to an alternative to avoid misconfiguration of that facility.

*Note: It is recommended not to use **SYSUSER** for processing transactions. It is recommended to use appropriate alternatives for your transactions.*

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).
MPL HTTP User in *Message Option*	Used for MPL authentication (older releases only). For customers using that old version, ensure *MPL HTTP Password* is also set correctly for the alternative.
XAI Authentication User in *Message Option*	Used for XAI authentication (older releases only). For customers using that old version, ensure *XAI Authentication Password* is also set correctly for the alternative.
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	Used for supporting IWS user proxy connections

Facility	Comments
ENVIRON.INI file	
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.

The **SYSUSER** can then be disabled using the following techniques:

- Remove **SYSUSER** from your configured security realm for authentication. This will prevent the user from authenticating.
- Optionally, disable the **SYSUSER** user object setting the *User Enable* attribute to **Disable**. This will disable the account from any authorized activity in the product.

Audit Facilities

The product has an inbuilt auditing 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.

About Audit

Auditing allows for the configurable tracking of changes to key data by online and Web Services users. The product has an inbuilt, configurable audit facility that tracks changes and allows authorized users to track changes on an individual user and change basis.

The use of the inbuilt audit facility is optional and can be enabled or disabled at any time.

Audit Configuration

Note: This section covers the soft table implementation of Auditing. There is also specialist Audit algorithm support on Business Objects 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.

The inbuilt Audit facility is configured at a table level. For each table you wish to enable audit upon the following needs to be configured:

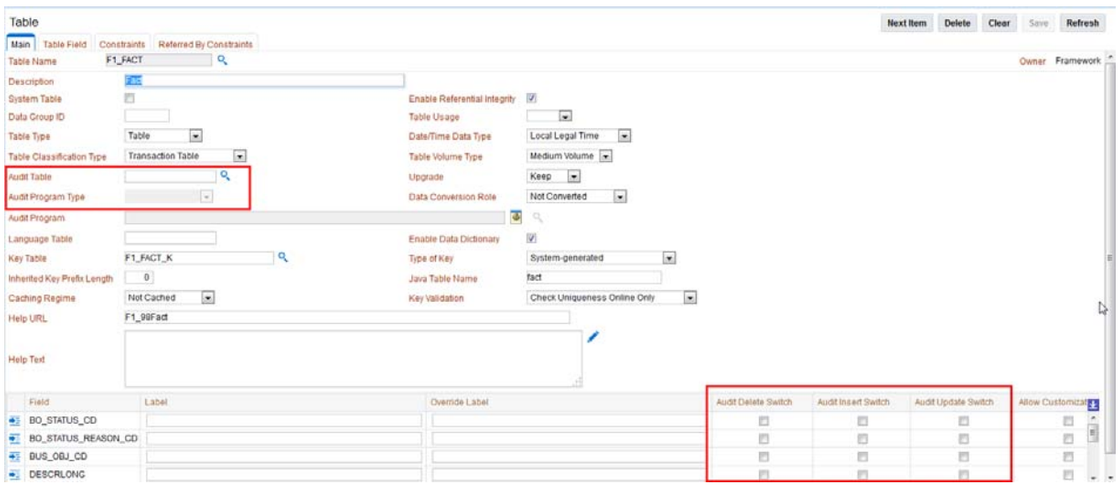
- **Audit Table** – To store the audit information a database table must be configured to hold the audit information. By default, the **`CI_AUDIT`** table can be used for this purpose. If a custom table is used to store the information it should have the same structure as **`CI_AUDIT`** for compatibility purposes.
- **Audit Program** – To process the audit information a class or program must be configured to record the audit information. By default a number of prebuilt 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 any fields configured to track auditing information.
  - **`com.splwg.base.domain.common.audit.ModifiedTableAuditor`** – This is an alternative to the **`DefaultTableAuditor`** but it will not audit inserts or deletes of empty string field data. For example, changes from null values to empty spaces and vice versa would not be 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 *Administration → T → Table* option and specify the table to enable auditing against. For example:

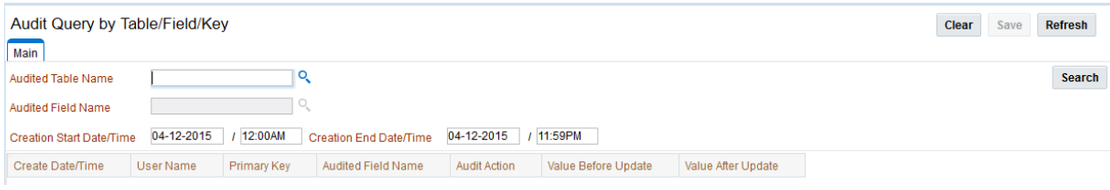


Specify the Audit Table, Audit Program (and associated type) and configure the Audit Switches on the fields you wish to track.

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 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 Table, Field and Key value to isolate changes. To access this query navigate to the *Administration → A → Audit Query By Table/Field/Key* option. For example:



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.

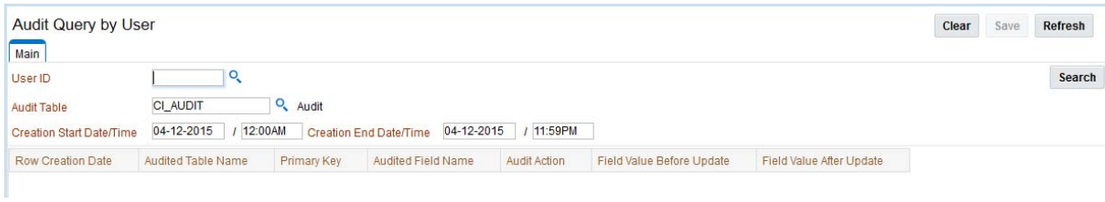
The query will return the following results:

- **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 *Administration → A → Audit Query By User* option. For example:



Specify any the following values for the filters:

- **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:

- **Row Creation Date** – Date and time the changes were made.
- **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 (i.e. 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. The concept of read auditing is different from the standard auditing as it is related to zones[10]. 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.

The information audited can be programmatically determined and which information is

[10] At the present time this parameter is available for **F1-DE**, **F1-DE-QUERY**, **F1-DE-SINGLE**, **F1-MAPDERV** and **F1-MAPEXPL** zone types only.

logged according to your requirements. Refer to the online zone help for descriptions and samples to configure 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.

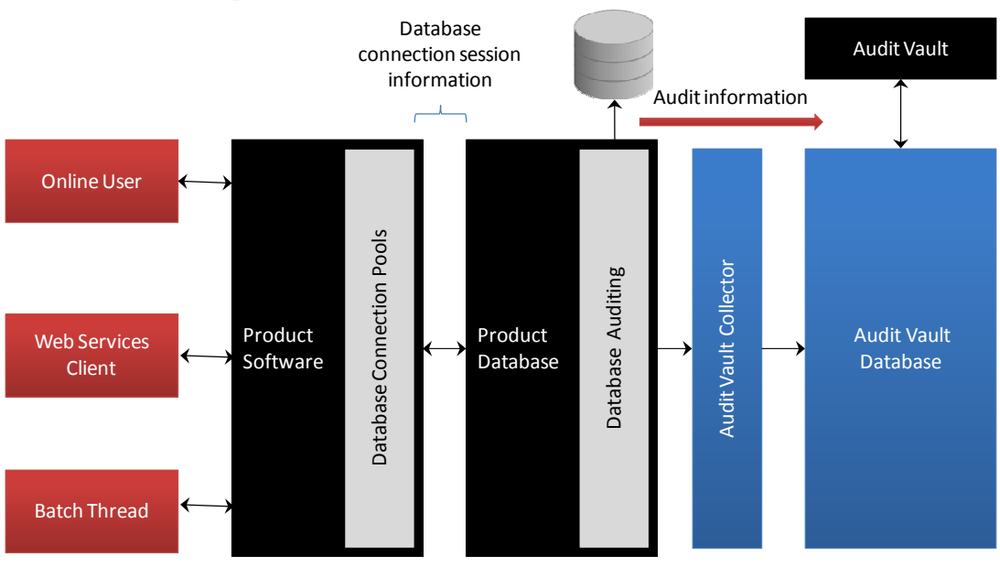
Integrating to Audit Vault

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 sufficient 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:

- **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 Data Collector 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.

Database Security

About 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*.

Database Users

The product 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[11].
- **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

The product 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,

[11] For customers on older versions of particular products this user was also used for the ConfigLab component.

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 product 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 of 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.

Note: For product tables with large amounts of data it is recommended to use the NOMAC feature to save disk space.

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 product 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.

Security Integration

About Security Integration

Whilst the product provides a set of security facilities natively or via the JEE Web Application Server, it is possible to augment the security with additional security features or security products.

LDAP Integration

By default, Oracle WebLogic includes an internal security repository that uses the Lightweight Directory Access Protocol (LDAP) to provide authentication facilities[12]. 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:

- The JEE Web Application Server must be configured to use the external LDAP security source for authentication. Refer to the documentation provided with the JEE Web Application Server 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 the JEE Web Application Server. Refer to the documentation provided with the JEE Web Application Server for more details.

Single Sign On Integration

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

The JEE Web Application Server 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.

[12] It also provides authorization services but these are not typically utilized by the product.

Kerberos Support

Single Sign-On (SSO) with Microsoft clients allows cross-platform authentication between Web applications running in the JEE Web Application Server 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:

- **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.

Keystore and Truststore Support

The Oracle Utilities Application Framework supports the ability to store cryptographic keys and/or certificates. The keystore is used to encrypt and decrypt data such as passwords and for the data encryption component of the Oracle Utilities Application Framework. It is also possible to implement a trust store to ensure the integrity of certificates.

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.

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 **`128`**.
- **`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 **128**.
- **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:

Linux/UNIX

```
initialSetup.sh –k|-K
```

Windows

```
initialSetup.cmd –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 | -K** utility to recreate the keystore (**-k**) or 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-encrypyted to match the new settings using the process outlined in Synchronize Data Encrytion.

Synchronize Data Encryption

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

Note: The product should be shutdown 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.
- XML Application Integration legacy passwords for JDBC.
- XAI 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:

  Windows:

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

  Unix/Linux:

  ```
  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=<oldpassfile> -
Dcom.oracle.ouaf.system.old.keystore.type=<oldtype> -
Dcom.oracle.ouaf.system.old.keystore.alias=<oldalias> -
Dcom.oracle.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 has to be run once for each alias.

After running ***ChangeCryptographyKey****, you must run* ***<SPLEBASE>/bin/invokeDBUpdatePatch.cmd|.sh*** *to reset the database patching credentials as follows:*

- If you have not already done so, use the **splenviron** 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.

Windows:

```
<SPLEBASE>\bin\invokeDBUpdatePatch.cmd -b
```

Unix/Linux:

```
<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.cmd|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 the product supplies a default keystore and truststore it is possible to import existing keystores and/or 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** are 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.

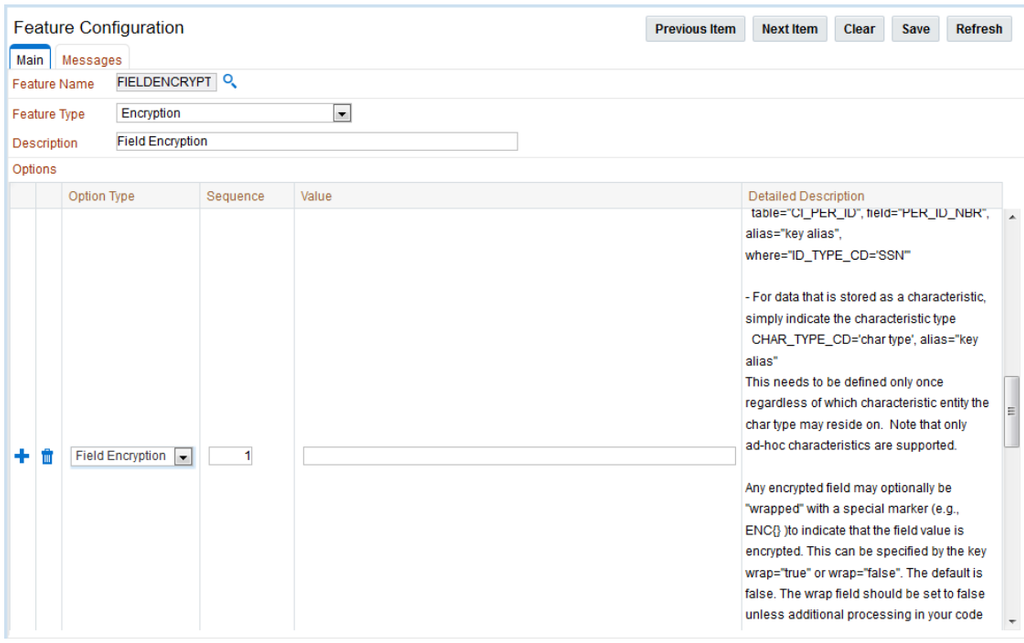
Encryption Feature Type

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

Overview

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 *Admin Menu* → *F* → *Feature Configuration* menu item. For example:



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",encryptedField="P
K_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.*

Web Services Security

About Web Services Security

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

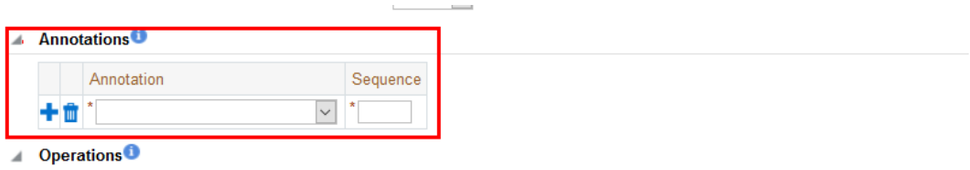
Note: Refer to the Web Services Best Practices (Doc Id: 2214375.1) for additional implementation advice for web services security.

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.
  - **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. For example:



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[13] 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*.

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.*

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 Web Services Manager for REST Services

By default the security used in REST services inherits the security of the security realm for the online or mobile channel. It is also possible to use Oracle Web Services Manager (OWSM) to provide additional security features. To use this facility the following must be configured:

- Set the *OWSM Protection for REST Services* to **`true`** using the **`configureEnv`** utility.

[13] Message level policy support is restricted to the whole message not within parts of the message.

- By default the following OWSM policies are then included in the **web.xml**:

  SSL **oracle/multi_token_over_ssl_rest_service_policy**

  Non-SSL **oracle/multi_token_rest_service_policy**

- It is possible to override these policies using WLST, Oracle WebLogic console, Oracle Fusion Middleware control or implementing a custom **web.xml** via a custom template. Refer to the *Server Administration Guide* and *Oracle WebLogic Administration* documentation for more information.

Note: Credentials are extracted via the Security Context user principal or the HTTP request remote user.

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.

Whitelist Support

About Whitelist Support

In the Oracle Cloud implementations of the product the use of whitelists is enforced to protect resources within the implementation. These whitelists can also apply to non-cloud implementations and in some cases can be extended to suit individual needs.

Note: Custom whitelists are not supported on Oracle Utilities SaaS Cloud Services.

URL Whitelist

Note: The URL whitelist 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 whitelist 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 whitelist.

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 whitelist 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 whitelist file
com.oracle.ouaf.customer.whitelist.file	Location and name of custom whitelist file

The format of the whitelist 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** etc
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 Whitelists

It is possible to establish a custom whitelist to implement URL whitelisting for non-cloud implementations. The format of the whitelist is the same as the above whitelist and this will augment and enhance the existing whitelist (if present). To support a custom whitelist 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 whitelist located as indicated in the **com.oracle.ouaf.whitelist.file** parameter. Ensure the custom file is in the same location as this whitelist 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 whitelist.
  `com.oracle.ouaf.customer.whitelist.file=<filename>`
- Execute the **initialSetup[.sh]** to implement the new setting.

SQL Whitelist

Note: Custom SQL Whitelists are not supported at this time.

Note: Sites not wanting to use this SQL function whitelist 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 whitelist is implemented as a Managed Content object named **F1-SQLFunctionWhiteList**. This whitelist is not changeable and lists the supported functions that are whitelisted for use. Any function used that is not used in this whitelist, if the facility is enabled, will generate a runtime error when the SQL is executed.

HTML WhiteList

The HTML used in UI Maps can be limited using a HTML whitelist. This product managed whitelist lists the valid HTML tags that can be used in the HTML objects. The whitelist is implemented as a Managed Content whitelist 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 Whitelist

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

To implement a custom HTML whitelist, copy the **F1-HTMLWhiteList** Managed Content to **CM-HTMLWhiteList** Managed Content and extend the whitelist.

*Note: Do not remove any tags that exist within **F1-HTMLWhiteList** in your custom whitelist. This may cause unexpected behavior across base UI Maps.*

*Note: Any upgrades to **F1-HTMLWhiteList** must be also manually reflected in **CM-HTMLWhiteList** for any subsequent upgrade.*

Groovy Whitelist

Note: The current implementation of the Groovy Whitelist 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 whitelisted to exclude parts of the language not appropriate for cloud implementations. The whitelist 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.

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.

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 a number of security scenarios:

- If the domain security realm uses a number of 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 Admin 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.

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.

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.
3rd Party Identity Provider (IdP)	-	This document uses Shibboleth as an example only[14]. Any 3rd party provider supporting SAML can be used. Refer to the Installation Reference for Oracle WebLogic and Oracle Interoperability.

Process Flow

When an user logins into the product 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:

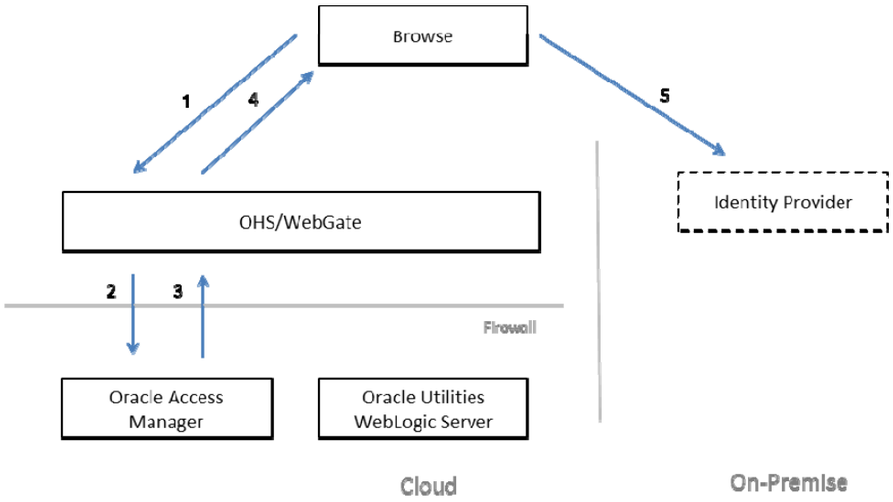
1. The user logs into the product using the provided login to the proxy (OHS/Webgate) which houses the access rules and detects that the user has not been authenticated yet.
2. Oracle Access Manager is called to determine that validation of the authentication is

[14] This is not a recommendation from Oracle for implementing this capability. Shibboleth is recognized as a reference infrastructure.

the responsibility of the external Identity Provider (IdP). It formats a SAML 2.0 request to be sent to the Identity Provider.

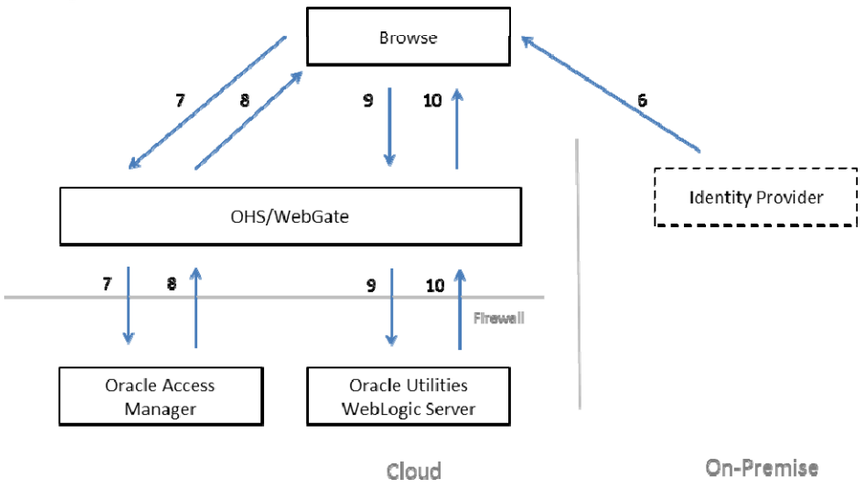
3. It responds to the user via the product as a redirect.
4. The browser redirects the request to the Identity Provider configured by Access Manager.
5. The SAML 2.0 request is sent to the Identity Provider for processing.

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.
- Gateway Configuration - Configure your proxy servers to accept requests and, optionally, support reverse proxy (to detect the client connection).
- Configure Access Management – Configure the access management solution with the details of the Identity Provider.
- Configure JIT Provisioning – 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 Agents – In a hybrid solution agents need to be configured to enable communication across partners.
- Ensure Agents are consistent – As agents need to communicate effectively ensure that the agent configurations match.
- Ensure Authentication Policies exist in domain – The domains must be configured with the linking policies to link all the components together.
- Communicate the SAML message format – 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 the product 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.

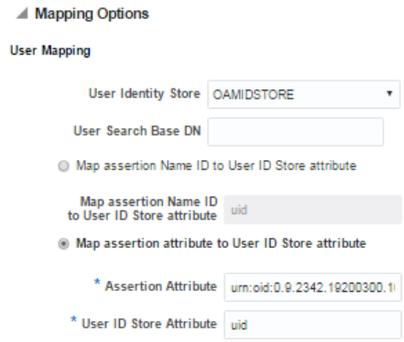
Define Identity Provider Partner in Oracle Access Manager

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.

Ensure that the following is configured as a minimum[15]:

- 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 logins into the solution for the first time, the user exists in the IdP and the product *User* object but 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.

[15] Using the Federation → Launch Pad → Service Provider Management → Identity Provider Partner option.

Configure the credentials for access[16] 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 product must be registered and configured.

For use with the product the following must be configured at a minimum:

- Set the Security to *Simple* (or *Cert*[17]).
- 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.

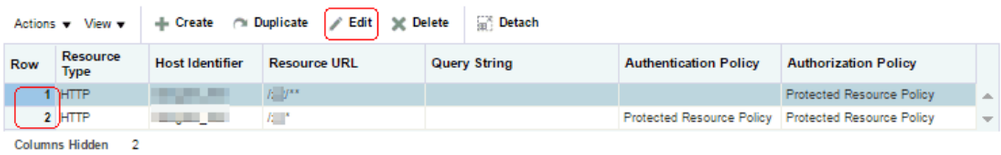
Copy WebGate Agent Configuration to OHS/WebGate

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 **`/spl*`**) as a *Protected Resource Policy*.
- Repeat the above step for subdirectories under the context, for example, **`/spl/**`**.



[16] Set the *Bind DN* to the connection group.

[17] For two way certificate implementations.

- 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 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 a Authentication Provider as per the Configuring an Oracle Access Manager 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

The product ships a Custom Authenticator 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 Service 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 Authentication Provider 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 product. 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[18] 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.

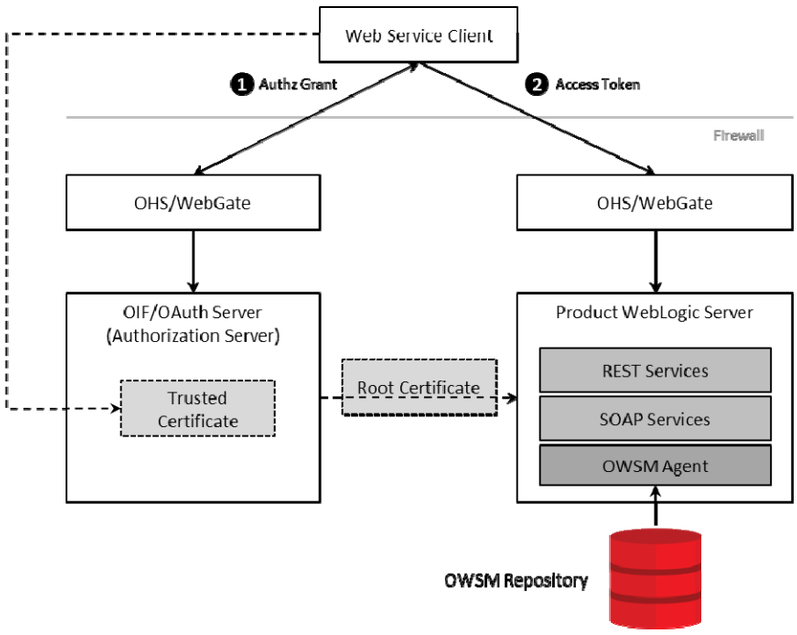
Process Flow

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:

[18] This is used for the Mobile Framework



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.

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

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

- 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[19]. 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`**.

[19] This is so ensure that the token is generated correctly for the Web Services when using a JWT token.

<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[20] into a temporary JKS keystore. For example[21]:

```
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

- **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
```

[20] 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 Using Third Party CA Signed Certificates to for more information.

[21] This example is for non-production only. Alter the command for alternative certificates.

```
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**

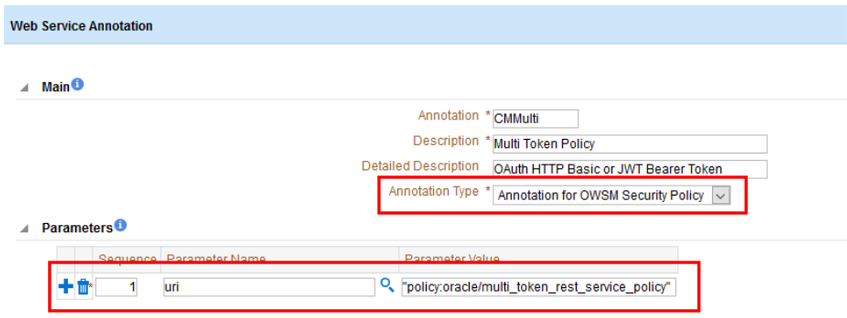
Use Oracle Web Service Manager Policies

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.

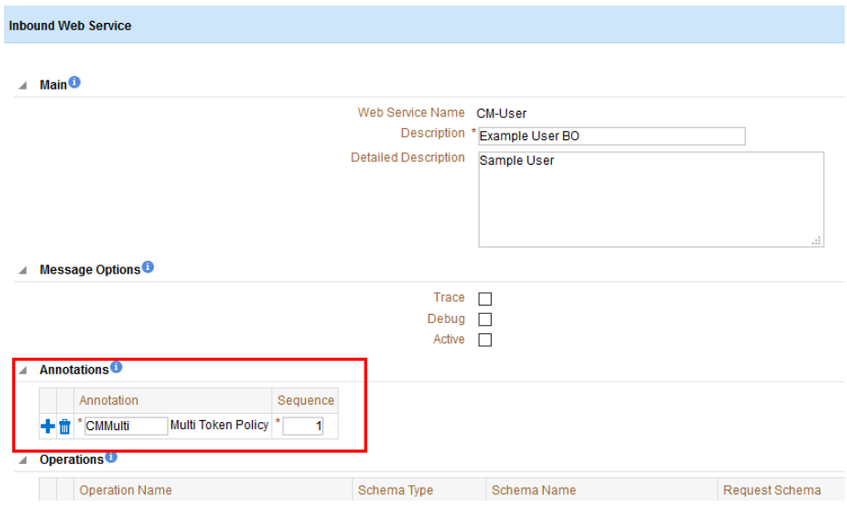
Individual Web Service Policies

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. For example:



- Attach the Web Service Annotation to the relevant services. For example:



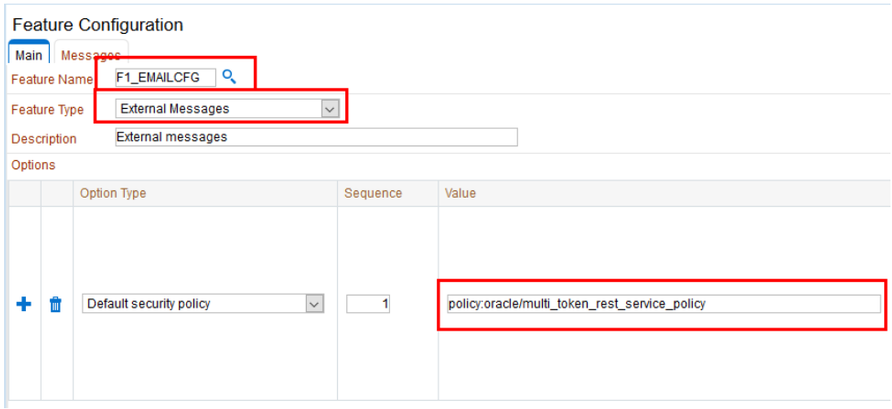
- Deploy/Redeploy the Inbound Web Services to implement the policy. Refer to the Server Administration Guide for more information.

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. For example:



- 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 that section for initial setup instructions.

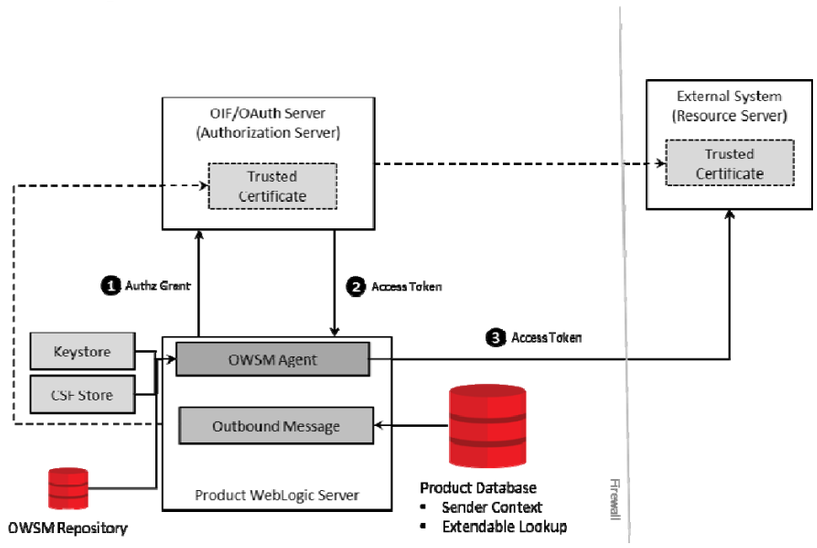
Outbound Messages allows product transactions to send information out of the product via technology connectors in synchronous and asynchronous modes.

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 must 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**)

Securing JNDI Access

Overview

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.

Securing Product Access

The JNDI registers all the Java EE 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 allocated 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.

Resource	Role/Group
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:

- Setup 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.