Oracle® Retail Merchandising

Security Guide

Release 14.0

E41056-01

December 2013

ORACLE®

Oracle Retail Merchandising Security Guide, Release 14.0

E41056-01

Primary Author: Seema Kamat

acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.

This documentation is in preproduction status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your beta trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Software License and Service Agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

Contents

2 Post Installation of Retail Infrastructure in Database

3 Post Installation of Retail Infrastructure in WebLogic

4 Installing the Merchandise Operations Management Security Applications

5 Troubleshooting

6 Importing Topology Certificate

7 Using Self Signed Certificates

Part V Oracle Retail Allocation

20 Setting up Functional Security

Part VI Active Retail Intelligence (ARI)

21 Security Considerations for Active Retail Intelligence (ARI)

List of Examples

List of Figures

List of Tables

Send Us Your Comments

Oracle Retail Merchandising Security Guide, Release 14.0

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

> **Note:** Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the Online Documentation available on the Oracle Technology Network Web site. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at http://www.oracle.com.

Preface

This guide serves as a guide for administrators, developers, and system integrators who securely administer, customize, and integrate Oracle Retail Merchandising Operations Management (MOM) Suite applications. Installation and configuration for each product are covered in more detail in the each product's Installation Guide.

Audience

This document is intended for administrators, developers, and system integrators who perform the following functions:

- Document specific security features and configuration details for the Oracle Retail MOM Suite products, in order to facilitate and support the secure operation of the Oracle Retail product and any external compliance standards.
- Guide administrators, developers, and system integrators on secure product implementation, integration, and administration. Functional and technical description of the problem (include business impact)

It is assumed that the readers have general knowledge of administering the underlying technologies and the application.

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 information, see the following documents:

- *Oracle® Retail Merchandising System Installation Guide*
- *Oracle® Retail Merchandising System Operations Guide*
- *Oracle® Retail Merchandising System Reports User Guide*

- *Oracle® Retail Merchandising System User Guide and Online Help*
- *Oracle® Retail Merchandising System Data Access Schema Data Model*
- *Oracle® Retail Merchandising System Release Notes*
- *Oracle® Retail Merchandising System Custom Flex Attribute Solution Implementation Guide*
- *Oracle® Retail Trade Management User Guide and Online Help*
- *Oracle® Retail Sales Audit User Guide and Online Help*
- *Oracle® Retail Merchandising Batch Schedule*
- *Oracle® Retail Merchandising System Data Model*
- *Oracle® Retail Merchandising Implementation Guide*
- *Oracle® Retail Merchandising Data Conversion Operations Guide*
- *Oracle® POS Suite/Merchandising Operations Management Implementation Guide*
- *Oracle® Retail Enterprise Integration Guide*

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

`https://support.oracle.com`

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 14.0) or a later patch release (for example, 14.0.1). If you are installing the base release, additional patch, and bundled hot fix releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch and bundled hot fix releases can contain critical information related to the base release, as well as information about code changes since the base release.

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times not be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Technology Network Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

This process will prevent delays in making critical corrections available to customers. For the customer, it means that before you begin installation, you must verify that you have the most recent version of the Oracle Retail documentation set. Oracle Retail documentation is available on the Oracle Technology Network at the following URL:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of a document with part number E123456-01.

If a more recent version of a document is available, that version supersedes all previous versions.

Oracle Retail Documentation on the Oracle Technology Network

Documentation is packaged with each Oracle Retail product release. Oracle Retail product documentation is also available on the following Web site:

http://www.oracle.com/technology/documentation/oracle-retail-100266.html

(Data Model documents are not available through Oracle Technology Network. These documents are packaged with released code, or you can obtain them through My Oracle Support.)

Documentation should be available on this Web site within a month after a product release.

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.

Part I

Oracle Retail Applications

The following chapters provide guidance for administrators, developers, and system integrators who securely administer, customize, and integrate the Oracle Retail Applications.

Part I contains the following chapters:

1

Pre-installation of Retail Infrastructure in WebLogic

Oracle Retail applications are primarily deployed in Oracle WebLogic server as Middleware tier. Java and forms based applications rely upon Middleware infrastructure for complete security apart from application specific security features.

This chapter describes the pre-installation steps for secured setup of Oracle Retail infrastructure in WebLogic.

The following topics are covered in this chapter:

Pre-installation - Steps for Secured Setup of Oracle Retail Infrastructure in WebLogic

Secured Sockets Layer (SSL) protocol allows client-server applications to communicate across a network in a secured channel. Client and server should both decide to use SSL to communicate secured information like user credentials or any other secured information.

WebLogic Server supports SSL on a dedicated listen port. Oracle Forms are configured to use SSL as well. To establish an SSL connection, a Web browser connects to WebLogic Server by supplying the SSL port and the Hypertext Transfer Protocol (HTTPs) protocol in the connection URL.

For example: https://myserver:7002

Retail Merchandising System (RMS) setup is supported in WebLogic in secured mode. For enterprise deployment, it is recommended to use SSL certificates signed by certificate authorities.

Note: You need to obtain a separate signed SSL certificates for each host where application is being deployed.

The Security Guide focuses on securing Oracle Retail Applications in single node setup and not on applications deployed on clusters.

Certificate Authority

Certificate Authority or Certification Authority (CA) is an organization which provides digital certificates to entities and acts as trusted third party. Certificates issued by the commercial CAs are automatically trusted by most of the web browsers, devices, and applications. It is recommended to have certificates obtained from a trusted CA or commercial CAs to ensure better security.

Obtaining an SSL Certificate and Setting up a Keystore

Note: SSL certificates are used to contain public keys. With each public key there is an associated private key. It is critically important to protect access to the private key. Otherwise, the SSL messages may be decrypted by anyone intercepting the communications.

Perform the following steps to obtain an SSL certificate and setting up a keystore:

1. Obtain an identity (private key and digital certificates) and trust (certificates of trusted certificate authorities) for WebLogic Server.
2. Use the digital certificates, private keys, and trusted CA certificates provided by the WebLogic Server kit, the CertGen utility, Sun Microsystem's keytool utility, or a reputed vendor such as Entrust or Verisign to perform the following steps:
    1. Set appropriate JAVA_HOME and PATH to java, as shown in the following example:

        ```
        export JAVA_HOME=/u00/webadmin/product/jdk
        export PATH=$JAVA_HOME/bin:$PATH
        ```

    2. Create a new keystore.

        keytool -genkey -keyalg RSA -keysize 2048 -keystore <keystore> -alias <alias>

        For example:

        ```
        keytool -genkey -keyalg RSA -keysize 2048 -keystore hostname.keystore
        -alias hostname
        ```

    3. Generate the signing request.

        keytool -certreq -keyalg RSA -file <certificate request file> -keystore <keystore> -alias <alias>

        For example:

        ```
        keytool -certreq -keyalg RSA -file hostname.csr -keystore hostname.keystore
        -alias hostname
        ```

    4. Submit the certificate request to CA.

3. Store the identity and trust.

   Private keys and trusted CA certificates which specify identity and trust are stored in a keystore.

   In the following examples the same keystore to store all certificates are used:

   1. Import the root certificate into the keystore as shown in the following example:

      ```
      keytool -import -trustcacerts -alias  verisignclass3g3ca -file Primary.pem
      -keystore  hostname.keystore
      ```

      A root certificate is either an unsigned public key certificate or a self-signed certificate that identifies the Root CA.

   2. Import the intermediary certificate (if required) into the keystore as shown in the following example:

      ```
      keytool -import -trustcacerts -alias oracleclass3g3ca -file Secondary.pem
      -keystore  hostname.keystore
      ```

   3. Import the received signed certificate for this request into the keystore as shown in the following example:

      ```
      keytool -import -trustcacerts -alias hostname -file cert.cer -keystore
      hostname.keystore
      ```

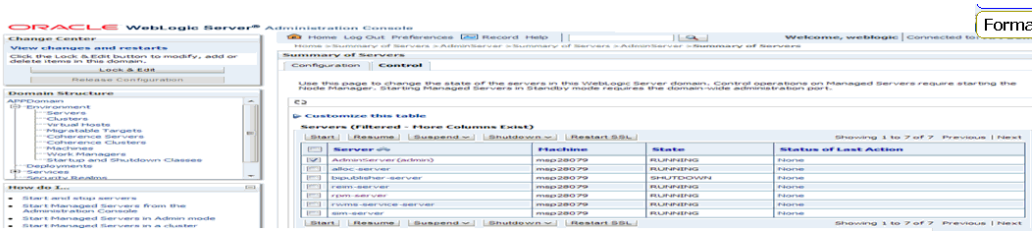
Creating a WebLogic Domain

WebLogic domain is created for Oracle Retail Applications as part of the installation. Different domains are created in different hosts for different applications in situations where applications are being managed by different users or deployed on different hosts. Once the domains are created, you need to enable the SSL ports if not done already.

Perform the following steps to enable the SSL:

1. Log in to WebLogic console using Administrator user. For example, weblogic.
2. Navigate to <Domain> > Environment > Servers > < Servername> > Configuration > General tab.
3. Click **Lock & Edit**.
4. Select **SSL Listen Port Enabled** and assign the port number.
5. Click **Save** and **Activate Changes**.
6. Restart SSL to enable the changes.

Figure 1–1 Restarting the Admin Server



Configuring the Application Server for SSL

Perform the following steps to configure the Application Server for SSL:

1. Configure the identity and trust keystores for WebLogic Server in the WebLogic Server Administration Console.
    1. In the Change Center of the Administration Console, click **Lock & Edit**.
    2. In the left pane of the Console, expand **Environment** and select **Servers**.
    3. Click the name of the server for which you want to configure the identity and trust keystores as shown in the following example:

        ```
        WLS_FORMS is for Forms server
        ```
    4. Select **Configuration**, then select **Keystores**.
    5. In the **Keystores** field, select the method for storing and managing private keys/digital certificate pairs and trusted CA certificates.

        The following options are available:

        - **Demo Identity and Demo Trust** - The demonstration identity and trust keystores, located in the BEA_HOME\server\lib directory and the Java Development Kit (JDK) cacerts keystore, are configured by default. You need to use for development purpose only.

        - **Custom Identity and Java Standard Trust** - A keystore you create and the trusted CAs defined in the cacerts file in the JAVA_HOME\jre\lib\security directory.

        - **Custom Identity and Custom Trust [Recommended]** - An Identity and trust keystores you create.

        - **Custom Identity and Command Line Trust**: An identity keystore you create and command-line arguments that specify the location of the trust keystore.
    6. Select **Custom Identity** and **Custom Trust**.
    7. In the **Identity** section, define the following attributes for the identity keystore:

        - **Custom Identity Keystore** - This is the fully qualified path to the identity keystore.

        - Custom Identity Keystore Type - This is the type of the keystore. Generally, this attribute is Java KeyStore (JKS); if it is left blank, it defaults to JKS.

        - Custom Identity Keystore Passphrase - This is the password you must enter when reading or writing to the keystore. This attribute is optional or required depending on the type of keystore. All keystores require the passphrase in order to write to the keystore. However, some keystores do not require the passphrase to read from the keystore. WebLogic Server only reads from the keystore so whether or not you define this property depends on the requirements of the keystore.
    8. In the **Trust** section, define properties for the trust keystore.

        If you choose **Java Standard Trust** as your keystore, specify the password defined when creating the keystore.
    9. Confirm the password.

        If you choose **Custom Trust [Recommended]** define the following attributes:

        - **Custom Trust Keystore** - This is the fully qualified path to the trust keystore.

        - **Custom Trust Keystore Type** - This is the type of the keystore. Generally, this attribute is JKS; if it is left blank, it defaults to JKS.

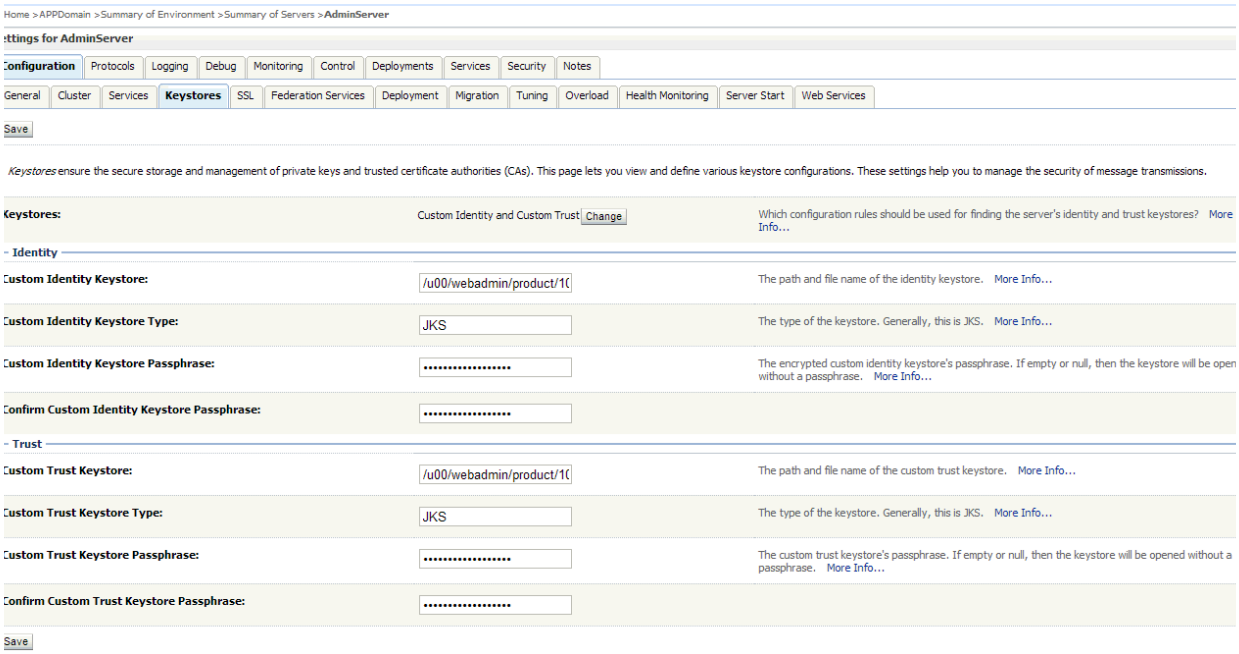
- **Custom Trust Keystore Passphrase** - This is the password that you need to enter when reading or writing to the keystore. This attribute is optional or required depending on the type of keystore. All keystores require the passphrase in order to write to the keystore. However, some keystores do not require the passphrase to read from the keystore. WebLogic Server only reads from the keystore, so whether or not you define this property depends on the requirements of the keystore.

10. Click **Save**.
11. To activate these changes, in the Change Center of the Administration Console, click **Activate Changes**.

Note: Not all changes take effect immediately, some require a restart.

Figure 1–2 shows how to configure the Application Server for SSL.

Figure 1–2 Configuring the Identity and Trust Keystores for WebLogic Server



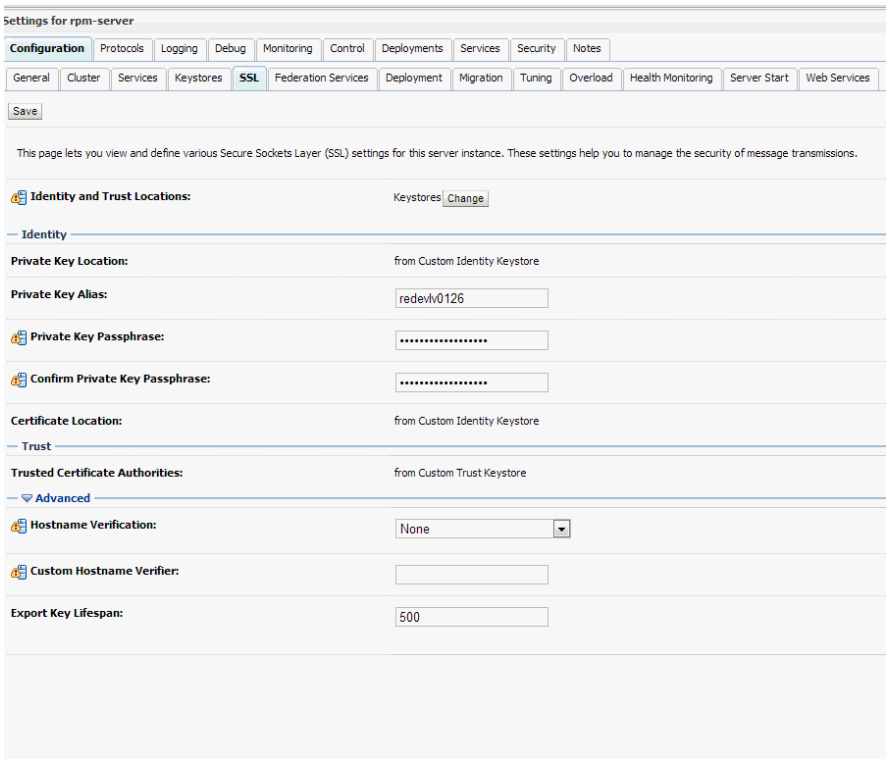
For more information on configuring Keystores, see the *Administration Console Online Help*.

2. Set SSL configuration options for the private key alias and password in the WebLogic Server Administration Console.
    1. In the Change Center of the Administration Console, click **Lock & Edit**.
    2. In the left pane of the Console, expand **Environment** and select **Servers**.
    3. Click the name of the server for which you want to configure the identity and trust keystores.
    4. Select **Configuration**, then select **SSL**.
    5. In the **Identity and Trust Locations**, the **Keystore** is displayed by default.

6. In the **Private Key Alias**, type the string alias that is used to store and retrieve the server's private key.
7. In the **Private Key Passphrase**, provide the keystore attribute that defines the passphrase used to retrieve the server's private key.
8. Save the changes.
9. Click **Advanced** section of SSL tab.
10. In the **Hostname Verification**, select **None**.

    This specifies to ignore the installed implementation of the WebLogic.security.SSL.HostnameVerifier interface (this interface is generally used when this server is acting as a client to another application server).
11. Save the changes.

Figure 1–3 Configuring SSL



For more information on configuring SSL, see the section *Configure SSL* in the *Administration Console Online Help*.

All the server SSL attributes are dynamic; when modified through the Console. They cause the corresponding SSL server or channel SSL server to restart and use the new settings for new connections. Old connections will continue to run with the old configuration. You must reboot WebLogic Server to ensure that all the SSL connections exist according to the specified configuration.

Use the **Restart SSL** button on the **Control**: Start/Stop page to restart the SSL server when changes are made to the keystore files. You have to apply the same for subsequent connections without rebooting WebLogic Server.

Upon restart you can see the following similar entries in the log:

```
<Mar 11, 2013 5:18:27 AM CDT> <Notice> <WebLogicServer> <BEA-000365> <Server
state changed to RESUMING>
<Mar 11, 2013 5:18:27 AM CDT> <Notice> <Server> <BEA-002613> <Channel
"DefaultSecure" is now ing on 10.141.15.214:57002 for protocols iiops, t3s,
ldaps, https.>
<Mar 11, 2013 5:18:27 AM CDT> <Notice> <Server> <BEA-002613> <Channel
"DefaultSecure[1]" is now ing on 127.0.0.1:57002 for protocols iiops, t3s,
ldaps, https.>
<Mar 11, 2013 5:18:27 AM CDT> <Notice> <WebLogicServer> <BEA-000329> <Started
WebLogic Admin Server "AdminServer" for domain "APPDomain" running in
Production Mode>
<Mar 11, 2013 5:18:27 AM CDT> <Notice> <WebLogicServer> <BEA-000365> <Server
state changed to RUNNING>
<Mar 11, 2013 5:18:27 AM CDT> <Notice> <WebLogicServer> <BEA-000360> <Server
started in RUNNING mode>
```

Note: For complete security of the WebLogic Server, it is recommended to secure both Administration as well the Managed Server where application is being deployed. You can choose to disable the non-SSL ports (HTTP). It is recommended to secure the Node Manager.

The steps to secure Node Manager is provided in the following section.

Configuring WebLogic Scripts if Admin Server is Secured

Perform the following steps to configure the WebLogic scripts if Admin Server is secured:

1. Update the WebLogic startup/shutdown scripts with secured port and protocol to start/stop services.
2. Backup and update the following files in <DOMAIN_HOME>/bin with correct Admin server urls:

   startManagedWebLogic.sh: echo "$1 managedserver1 http://apphost1:7001"

   stopManagedWebLogic.sh: echo "ADMIN_URL defaults to t3://apphost1:7001 if not set as an environment variable or the second command-line parameter."

   stopManagedWebLogic.sh: echo "$1 managedserver1 t3://apphost1:7001 WebLogic WebLogic"

   stopManagedWebLogic.sh: ADMIN_URL="t3://apphost1:7001"

   stopWebLogic.sh: ADMIN_URL="t3://apphost1:7001"
3. Change the URLs as follows:

   t3s://apphost1:7002

   https://apphost1:7002

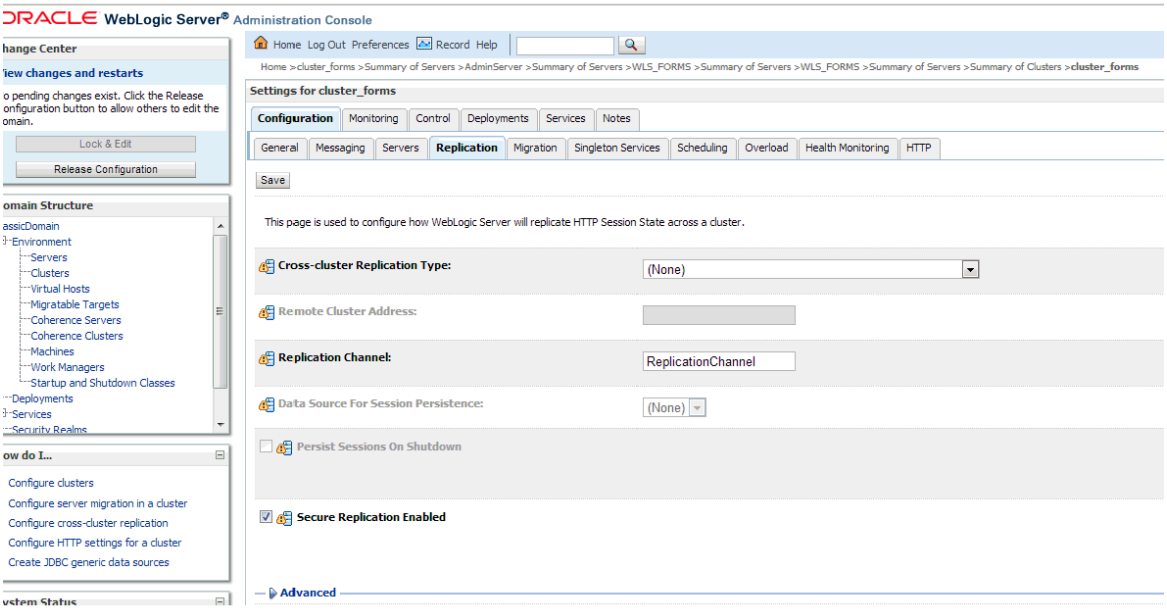
Additional Configuration for WLS_FORMS (For forms server)

Perform the following steps for WebLogic forms:

1. Log in to **WebLogic Console**. Select **Environment** > **Clusters** > **cluster_forms**, then select **Configuration** > **Replication**.
2. Select **Secure Replication Enabled**.

3. Start the **WLS_FORMS** Managed server.

Figure 1–4 WebLogic Server Forms



Adding Certificate to the JDK Keystore for Installer

You will need the Oracle Retail Application installer to run Java. In situations where Administration Server is secured using signed certificate, the Java keystore through which the installer is launched must have the certificate installed.

In case the installer is being run using JDK deployed at location /u00/webadmin/product/jdk, follow the steps as shown in Example 1–1.

Example 1–1 Adding certificate to the JDK keystore for Installer

```
apphost1:[10.3.6_apps] /u00/webadmin/ssl> keytool -import -trustcacerts -alias
apphost1 -file /u00/webadmin/ssl/apphost1.cer -keystore
/u00/webadmin/product/jdk/jre/lib/security/cacerts
Enter keystore password:
Certificate was added to keystore
apphost1:[10.3.6_apps] /u00/webadmin/ssl>
```

Enforcing Stronger Encryption in WebLogic

It is recommended to use a stronger encryption protocol in your production environment.

See the following sections to enable the latest SSL and cipher suites.

SSL protocol version configuration

In a production environment, Oracle recommends Transport Layer Security (TLS) Version 1.1, or higher for sending and receiving messages in an SSL connection.

To control the minimum versions of SSL Version 3.0 and TLS Version 1 that are enabled for SSL connections, do the following:

- Set the **WebLogic.security.SSL.minimumProtocolVersion=protocol** system property as an option in the command line that starts WebLogic Server.

  This system property accepts one of the following values for protocol:

Figure 1–5 Values for Protocol of System Property

Value	Description
`SSLv3`	Specifies SSL V3.0 as the minimum protocol version enabled in SSL connections.
`TLSv1`	Specifies TLS V1.0 as the minimum protocol version enabled in SSL connections.
`TLSvx.y`	Specifies TLS V`x.y` as the minimum protocol version enabled in SSL connections, where: • `x` is an integer between 1 and 9, inclusive • `y` is an integer between 0 and 9, inclusive For example, `TLSv1.2`.

- Set the following property in startup parameters in WebLogic Managed server for enabling the higher protocol:

  DWebLogic.security.SSL.minimumProtocolVersion=TLSv1.1

> **Note:** In case protocol is set for Managed servers, the same should be set for Administration server. Ensure that all the managed servers are down when making changes to the Administration server for setting up the protocol. It is recommended to set the properties in Administration server and then the Managed server.

Upgrading JDK to Use Java Cryptography Extension

You need to install the unlimited encryption Java Cryptography Extension (JCE) policy, if you want to use the strongest Cipher suite (256 bit encryption) AES_256 (**TLS_RSA_WITH_AES_256_CBC_SHA**). It is dependent on the Java Development Kit (JDK) version.

Using the following URL, download and install the JCE Unlimited Strength Jurisdiction Policy Files that correspond to the version of your JDK:

`http://www.oracle.com/technetwork/java/javase/downloads/index.html`

For JDK 7, download from the following URL:

`http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html` and replace the files in JDK/jre/lib/security directory

> **Note:** Restart the entire WebLogic instance using the JDK to enable changes to take effect once the JCE has been installed.

Enabling Cipher in WebLogic SSL Configuration

Configure the <ciphersuite> element in the <ssl> element in the <DOMAIN_HOME>\server\config\config.xml file in order to enable the specific Cipher Suite to use as follows:

> **Note:** You need to ensure that the tag <ciphersuite> is added immediately after tab <enabled>.

```
<ssl>
<name>examplesServer</name>
```

```
<enabled>true</enabled>
<ciphersuite>TLS_RSA_WITH_AES_256_CBC_SHA</ciphersuite>
<-port>17002</-port>
...
</ssl>
```

Note: The above can be done using wlst script.

For more information, go to http://docs.oracle.com/cd/E24329_01/web.1211/e24422/ssl.htm#BABDAJJG. It is advisable to bring down the managed server prior to making the changes.

Securing Nodemanager with SSL Certificates

Perform the following steps for securing the Nodemanager with SSL certificates:

1. Navigate to **<BEA_HOME>/wlserver_10.3/common/nodemanager** and take a backup of nodemanager.properties.
2. Add the following similar entries to nodemanager.properties:

   KeyStores=CustomIdentityAndCustomTrust

   CustomIdentityKeyStoreFileName=/u00/webadmin/ssl/hostname.keystore

   CustomIdentityKeyStorePassPhrase=[password to keystore, this will get encrypted]

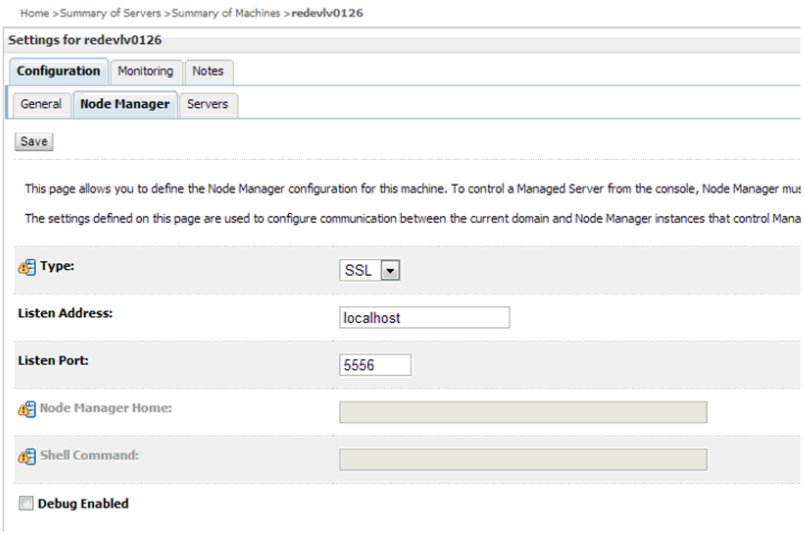
   CustomIdentityAlias=hostname

   CustomIdentityPrivateKeyPassPhrase=[password to keystore, this will get encrypted]

   CustomTrustKeyStoreFileName=/u00/webadmin/ssl/hostname.keystore

   SecureListener=true
3. Log in to **WebLogic console**, navigate to **Environment**, and then **Machines**.
4. Select the nodemanager created already and navigate to **Node Manager** tab.
5. In the Change Center, click **Lock & Edit**.
6. In the **Type** field, select **SSL** from the list.
7. Click **Save** and **Activate**.

Figure 1–6 Securing the Nodemanager



8. You need to bounce the entire WebLogic Domain for changes to take effect, after activating the changes.
9. You need to verify if the nodemanager is reachable in **Monitoring** tab after restart.

Using Secured Lightweight Directory Access Protocol (LDAP)

The Application can communicate with LDAP server on a secured port. It is recommended to use the secured LDAP server to protect user names and passwords from being sent in clear text on the network.

For information on Configuring Secure Sockets Layer (SSL), see the *Oracle Fusion Middleware Administration Guide*.

It is important to import the certificates used in LDAP server into the Java Runtime Environment (JRE) of the WebLogic server for SSL handshake, in case the secure LDAP is used for authentication.

For example:

1. Set JAVA_HOME and PATH to the JDK being used by WebLogic Domain.
2. Backup the JAVA_HOME/jre/lib/security/cacerts

   /u00/webadmin/product/jdk/jre/lib/security> cp -rp cacerts cacerts_ORIG
3. Import the Root and Intermediary (if required) certificates into the java keystore.

   /u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts -alias verisignclass3g3ca -file ~/ssl/Primary.pem -keystore cacerts

   /u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts -alias oracleclass3g3ca -file ~/ssl/Secondary.pem -keystore cacerts
4. Import the User certificate from LDAP server into the java keystore.

   /u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts -alias hostname -file ~/ssl/cert.cer -keystore cacerts

Note: The default password of JDK keystore is **changeit**.

The deployed application should be able to communicate with LDAP on SSL port after successful SSL Handshake.

Connecting from Forms Application to Secured Database

RMS and Oracle Retail Warehouse Management System (RWMS) connect to the database using the Transparent Network Substrate (TNS) Alias as setup in tnsnames.ora file available in the location mentioned in RMS or RWMS environment file created during installation. Example 1–2 refers to an RMS Forms environment file, but the same steps apply to RWMS.

Example 1–2 Identify TNS_ADMIN setting in environment file created during installation

```
$ grep TNS_ADMIN <WLS_HOME> /user_
projects/domains/ClassicDomain/config/fmwconfig/servers/WLS_
FORMS/applications/formsapp_11.1.2/config/develop/rmsFqa3.env
TNS_ADMIN=/u00/webadmin/product/10.3.X_FORMS/WLS/asinst_1/config
```

For secured setup, the TNS Alias inside tnsnames.ora should refer to the TCPS port for Secured Listener of the database.

Example 1–3 Referring TNS Alias inside tnsnames.ora to the TCPS port for Secured Listener of the database

```
qaols03_secure =
(DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = tcps)(host = dbhost1)(Port =
2484)))
(CONNECT_DATA = (SID = qaols03) (GLOBAL_NAME = qaols03)))
```

Enabling Access to Secured Database from Forms Oracle Home - Optional

You need to perform the following additional setup to connect to Oracle database on secured port (TCPs) from Forms Oracle Home:

1. Create wallet using orapki.

Note: A wallet is created using either orapki or mkstore utility. Forms installation provides orapki utility to create the wallet and is used for creation of wallet.

```
$ mkdir /u00/webadmin/product/10.3.X_FORMS/WLS/Oracle_FRHome1/network/wallet
$ cd /u00/webadmin/product/10.3.X_FORMS/WLS/Oracle_FRHome1/network/wallet

$  export JAVA_HOME=/u00/webadmin/product/jdk
$  export PATH=$JAVA_HOME/bin:$PATH

$  export ORACLE_HOME=/u00/webadmin/product/10.3.X_FORMS/WLS/Oracle_FRHome1
$  export PATH=$ORACLE_HOME/bin:$PATH
$  export PATH=/u00/webadmin/product/10.3.X_FORMS/WLS/oracle_common/bin:$PATH
$  orapki wallet create -wallet
/u00/webadmin/product/10.3.X_FORMS/WLS/Oracle_FRHome1/network/wallet/secured
-auto_login -pwd <wallet-pwd>
Oracle PKI Tool: Version 11.1.1.5.0
Copyright (c) 2004, 2011, Oracle and/or its affiliates. All rights reserved.
$  ls
```

```
cwallet.sso  ewallet.p12
```

2. Import the Signed certificates into the wallet.

Example 1–4 Importing all certificates into the wallet

```
$ orapki wallet jks_to_pkcs12 -wallet
/u00/webadmin/product/10.3.X_FORMS/WLS/Oracle_FRHome1/network/wallet/secured -pwd
<wallet-pwd> -keystore
/u00/webadmin/product/10.3.X_APPS/WLS/wlserver_10.3/server/lib/apphost1.keystore
-jkspwd <keystore-pwd>
Oracle PKI Tool: Version 11.1.1.5.0
Copyright (c) 2004, 2011, Oracle and/or its affiliates. All rights reserved.
```

For information on Oracle Wallet Manager and orapki, see *Fusion Middleware Administrator's Guide*.

3. Provide the wallet details in sqlnet.ora file.

> **Note:** You need to create a sqlnet.ora file with details of the wallet in $ORACLE_HOME/network/admin directory, if the file is not available.

Example 1–5 sqlnet.ora file

```
SQLNET.AUTHENTICATION_SERVICES=(TCPS,NTS)
SSL_CLIENT_AUTHENTICATION = TRUE
WALLET_LOCATION =
   (SOURCE =
     (METHOD = FILE)
     (METHOD_DATA = (DIRECTORY = /u00/webadmin/product/10.3.X_FORMS/WLS/Oracle_
FRHome1/network/wallet/secured))
)
```

4. Connect using sqlplus.

```
$ export ORACLE_HOME=/u00/webadmin/product/10.3.X_FORMS/WLS/Oracle_FRHome1
$ export PATH=$ORACLE_HOME/bin:$PATH

$ sqlplus rms01app@qaols03_secure

SQL*Plus: Release 11.1.0.7.0 - Production on Thu Mar 28 02:31:19 2013

Copyright (c) 1982, 2008, Oracle.  All rights reserved.

Enter password: *******

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.3.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL>
```

Webservice Security Policies

You need to configure the user credentials and other security related information at the service consumer and the app service provider layers, in order to provide end to end security between Web service consumer and provider.

The security policies certified by Oracle Retail are as follows:

1. Username Token over HTTPS - This security configuration is referred as Policy A in this document. This policy provides confidentiality due to the use of SSL, however it does not provide non-repudiation as nothing is signed.

   Wssp1.2-2007-Https-UsernameToken-Plain.xml

2. Message Protection - This security configuration is referred as Policy B in this document. This policy encrypts the messages itself, so SSL is not used. The sender also signs the messages, which provides non-repudiation of the messages. However, this policy is more complex to implement.

   - Wssp1.2-2007-Wss1.1-UsernameToken-Plain-EncryptedKey-Basic128
   - Wssp1.2-2007-EncryptBody
   - Wssp1.2-2007-SignBody

   Note:

   1. The web services are secured using WebLogic policies (as opposed to OWSM policies).
   2. If the application services are secured with any policy other than what is mentioned in this document or custom policies, the instructions in the document will not work.
   3. The security setup in the document does not address authorization. Authorization must be taken care by the individual application hosting the services.
   4. Policy B is not supported over HTTPS. So ensure that non SSL ports are enabled prior to applying Policy B.

Additional Pre-requisite for Oracle Retail Service Backbone (RSB) Security Policies

Perform the additional pre-requisites for Oracle Retail Service Backbone (RSB) security policies:

1. Create DB schema for OSB [PolicyA][PolicyB].
2. Ensure that <RSB_MDS> schema is created while running Repository Creation Utility (RCU) at <rcuHome>/bin/rcu.
3. Extend RSB Domain with OWSM Extention [PolicyA][PolicyB].
4. Ensure that OSB OWSM Extension-11.1.1.6 is selected, when RSBDomain is being created.

Advanced Infrastructure Security

Depending upon your security need for your production environment, infrastructure where Oracle Retail applications are deployed can be secured.

Ensure the following to secure complete protection of environment:

- Securing the WebLogic Server Host
- Securing Network Connections
- Securing your Database
- Securing the WebLogic Security Service
- Securing Applications

For more information on Ensuring the Security of Your Production Environment, see *Securing a Production Environment for Oracle WebLogic Server. 11g Release 1. (10.3.6) Guide*.

2

Post Installation of Retail Infrastructure in Database

Oracle Retail applications use the Oracle database as the backend data store for applications. In order to ensure complete environment security the database should be secured.

This chapter describes the post installation steps for secured setup of Retail infrastructure in the Database.

The following topics are covered in this chapter:

Configuring SSL Connections for Database Communications

Secure Sockets Layer (SSL) is the standard protocol for secure communications, providing mechanisms for data integrity and encryption. This can protect the messages sent and received by the database to applications or other clients, supporting secure authentication and messaging. Configuring SSL for databases requires configuration on both the server and clients, which include application servers.

This section covers the steps for securing Oracle Retail Application Clusters (RAC) database. Similar steps can be followed for single node installations also.

Configuring SSL on the Database Server

The following steps are one way to configure SSL communications on the database server:

1. Obtain an identity (private key and digital certificate) and trust (certificates of trusted certificate authorities) for the database server from a Certificate Authority.
2. Create a folder containing the wallet for storing the certificate information. For Real Application Cluster (RAC) systems, this directory can be shared by all nodes in the cluster for easier maintenance.

   mkdir /oracle/secure_wallet
3. Create a wallet in the path. For example,

   ```
   orapki wallet create -wallet /oracle/secure_wallet -auto_login
   ```

4. Import each trust chain certificate into the wallet as shown in the following example:

```
orapki wallet add -wallet /oracle/secure_wallet -trusted_cert -cert <trust
chain certificate>
```

5. Import the certificate into the wallet, as shown in the following example:

```
orapki wallet add -wallet /oracle/secure_wallet -user_cert -cert <certificate
file location>
```

6. Update the listener.ora by adding a TCPS protocol end-point first in the list of end points

```
LISTENER1=
  (DESCRIPTION=
     (ADDRESS=(PROTOCOL=tcps)(HOST=<dbserver name>)(PORT=2484))
     (ADDRESS=(PROTOCOL=tcp)(HOST=<dbserver name>)(PORT=1521)))
```

7. Update the listener.ora by adding the wallet location and disabling SSL authentication.

```
WALLET_LOCATION =
 (SOURCE=
  (METHOD=File)
  (METHOD_DATA=
   (DIRECTORY=wallet_location)))
SSL_CLIENT_AUTHENTICATION=FALSE
```

8. Update the sqlnet.ora with the same wallet location information and disabling SSL authentication.

```
WALLET_LOCATION =
 (SOURCE=
  (METHOD=File)
  (METHOD_DATA=
   (DIRECTORY=wallet_location)))
SSL_CLIENT_AUTHENTICATION=FALSE
```

9. Update the tnsnames.ora to configure a database alias using TCPS protocol for connections.

```
<dbname>_secure=
  (DESCRIPTION=
   (ADDRESS_LIST=
    (ADDRESS=(PROTOCOL=TCPS)(HOST=<dbserver>)(PORT=2484)))
   (CONNECT_DATA=(SERVICE_NAME=<dbname>)))
```

10. Restart the database listener to pick up listener.ora changes.
11. Verify the connections are successful to the new <dbname>_secure alias
12. At this point either the new secure alias can be used to connect to the database, or the regular alias can be modified to use TCPS protocol.
13. Export the identity certificate so that it can be imported on the client systems

    orapki wallet export -wallet /oracle/secure_wallet -dn <full dn of identity certificate> -cert <filename_to_create>

Configuring SSL on an Oracle Database Client

The following steps are one way to configure SSL communications on the database client:

1. Create a folder containing the wallet for storing the certificate information.

   mkdir /oracle/secure_wallet

2. Create a wallet in the path. For example,

orapki wallet create -wallet /oracle/secure_wallet -auto_login

3. Import each trust chain certificate into the wallet as shown in the following example:

   orapki wallet add -wallet /oracle/secure_wallet -trusted_cert -cert <trust chain certificate>

4. Import the identity certificate into the wallet, as shown in the following example:

   orapki wallet add -wallet /oracle/secure_wallet -trusted_cert -cert <certificate file location>

   Note: On the client the identity certificate is imported as a trusted certificate, whereas on the server it is imported as a user certificate.

5. Update the sqlnet.ora with the wallet location information and disabling SSL authentication.

   WALLET_LOCATION =

   (SOURCE=

   (METHOD=File)

   (METHOD_DATA=

   (DIRECTORY=wallet_location)))

   SSL_CLIENT_AUTHENTICATION=FALSE

6. Update the tnsnames.ora to configure a database alias using TCPS protocol for connections.

   <dbname>_secure=

   (DESCRIPTION=

   (ADDRESS_LIST=

   (ADDRESS=(PROTOCOL=TCPS)(HOST=<dbserver>)(PORT=2484)))

   (CONNECT_DATA=(SERVICE_NAME=<dbname>)))

7. Verify the connections are successful to the new <dbname>_secure alias.

8. At this point either the new secure alias can be used to connect to the database, or the regular alias can be modified to use TCPS protocol.

Configuring SSL on a Java Database Connectivity (JDBC) Thin Client

The following steps are one way to configure SSL communications for a Java Database Connectivity (JDBC) thin client:

1. Create a folder containing the keystore with the certificate information.

   mkdir /oracle/secure_jdbc

2. Create a keystore in the path. For example,

   ```
   keytool -genkey -alias jdbcwallet -keyalg RSA -keystore /oracle/secure_
   jdbc/truststore.jks -keysize 2048
   ```

3. Import the certificate into the trust store as shown in the following example:

   ```
   keytool -import -alias db_cert -keystore /oracle/secure_jdbc/truststore.jks
   -file <db certificate file>
   ```

4. JDBC clients can use the following URL format for JDBC connections:

```
jdbc:oracle:thin:@(DESCRIPTION= (ADDRESS= (PROTOCOL=tcps) (HOST=<dbserver>)
(PORT=2484)) (CONNECT_DATA= (SERVICE_NAME=<dbname>)))
```

5. You need to set the properties as shown in Table 2–1, either as system properties or as JDBC connection properties.

Table 2–1 Setting the Properties

Property	Value
javax.net.ssl.trustStore	Path and file name of trust store. For example, /oracle/secure_jdbc/truststore.jks
javax.net.ssl.trustStoreType	JKS
javax.net.ssl.trustStorePassword	Password for trust store

Configuring the Password Stores for Database User Accounts

Wallets can be used to protect sensitive information, including usernames and passwords for database connections. The Oracle Database client libraries have built-in support for retrieving credential information when connecting to databases. Oracle Retail applications utilize this functionality for non-interactive jobs such as batch programs so that they are able to connect to the database without exposing user and password information to other users on the same system.

For information on configuring wallets for database access, see the Appendix Setting Up Password Stores with Oracle Wallet in the product installation guide.

Configuring the Database Password Policies

Oracle Database includes robust functionality to enforce policies related to passwords such as minimum length, complexity, when it expires, number of invalid attempts, and so on. Oracle Retail recommends these policies are used to strengthen passwords and lock out accounts after failed attempts.

For example, to modify the default user profile to lock accounts after five failed login attempts, run the following commands as a database administrator:

1. Query the current settings of the default profile

   select resource_name,limit,resource_type from dba_profiles where profile='DEFAULT';

2. Alter the profile, if failed_login_attempts is set to unlimited:

   alter profile default limit FAILED_LOGIN_ATTEMPTS 5;

> **Note:** Many other profile settings are available for increased security. For more information, see the *Oracle Database Security Guide.*

Additional Information

For more information on the subjects covered in this section as well as information on other options that are available to strengthen database security, see the *Oracle Database Security Guide 11g Release 2*.

The Oracle Advanced Security Option provides industry standards-based solutions to solve enterprise computing security problems, including data encryption and strong authentication. Some of the capabilities discussed in this guide require licensing the Advanced Security Option.

For more information, see the *Oracle Database Advanced Security Administrator's Guide 11g Release 2*.

3

Post Installation of Retail Infrastructure in WebLogic

This chapter describes the post installation steps for secured setup of Oracle Retail infrastructure in WebLogic.

The following topics are covered in this chapter:

Retail Application Specific Post installation Steps for Security

See the following sections for steps to improve security after an Oracle Retail Application has been installed.

Batch Set Up for SSL Communication

Java batch programs communicate with Java applications deployed in WebLogic. For example, Oracle Retail Price Management (RPM) and Oracle Store Inventory Management (SIM). The communication needs to have SSL handshake with the deployed application. You need to import the SSL Certificates into the JAVA_HOME/jdk/jre/lib/security/cacerts keystore for successful running of the application batches.

Example 3–1 Importing certificates into JDK keystore

```
/u00/webadmin/product/jdk/jre/lib/security> cp -rp cacerts cacerts_ORIG

/u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts -alias
verisignclass3g3ca -file ~/ssl/Primary.pem -keystore cacerts

/u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts -alias
oracleclass3g3ca -file ~/ssl/Secondary.pem -keystore cacerts

/u00/webadmin/product/jdk/jre/lib/security> keytool -import -trustcacerts -alias
hostname -file ~/ssl/cert.cer -keystore cacerts
```

Note: The default password of JDK keystore is **changeit**.

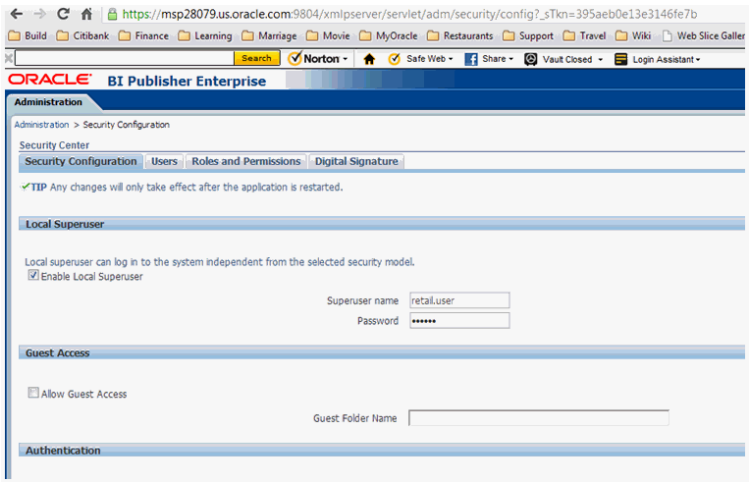
Oracle Business Intelligence (BI) Publisher - Disable Guest User - Optional

The guest account in Oracle Business Intelligence (BI) publisher is used for public facing reports that anyone can see. Disabling this account forces all users to supply their credentials before accessing any information. Disabling guest user enhances security of BI Publisher. However, application which requires guest user will have reporting feature which may cease to function after making this change. For example, RMS reports.

Perform the following steps to disable the guest user:

1. Log in to BI Publisher with user having Administrator privileges.
2. Navigate to Administration > Security Configuration.
3. Deselect **Allow Guest Access**.

Figure 3–1 Administration Window



4. Save and restart the BI Publisher instance.

RMS - Forms Timeout Setting - Optional

Oracle Forms can be configured to timeout based on user idle time.

You need to set the following parameters:

1. FORMS_TIMEOUT - This parameter is set RMS/RWMS env file created at <DOMAIN_HOME>/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_11.1.2/config directory

   The default value for forms timeout is 15 and Valid Values range from 3 to 1440 (1 day).

   This parameter specifies the amount of time in elapsed minutes before the Form Services process is terminated when there is no client communication with the Form Services. Client communication can come from the user doing some work, or from the Forms Client heartbeat if the user is not actively using the form.

2. HeartBeat - This parameter is set in formsweb.cfg file located in DOMAIN_HOME>/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_11.1.1/config directory.

   The default value for **HeartBeat** is 2 and Valid Values range from 1 to 1440 (1 day).

Example:

```
[rmsFqa3]
envfile=./develop/rmsFqa3.env
width=950
height=685
separateFrame=true
form=rtkstrt.fmx
lookAndFeel=Oracle
colorScheme=swan
archive=frmall.jar,icons.jar
imageBase=codebase
heartbeat=12
```

Note: For more information on the above parameters and additional options, see the Oracle Support Note: *Description List For Parameters Affect Timeout In Webforms [ID 549735.1].*

4

Installing the Merchandise Operations Management Security Applications

This chapter indicates the additional steps to be followed in conjunction with Oracle Retail Applications standard installation document. For more information on installing the Oracle Retail Applications, see the *Installation Guides* listed in the subsequent sections for complete steps to be followed.

This chapter covers the following topics of each Merchandise Operations Management (MOM) Applications:

- Installing the ReIM Application
- Installing the RPM Application
- Installing the RMS Application
- Installing the Allocation Application

Installing the ReIM Application

For information on steps related to the installation of ReIM in secured environment, see Chapter 10 and *Oracle Retail Invoice Matching Installation Guide*.

Installing the RPM Application

For information on steps related to the installation of RPM in secured environment, see Chapter 15 and *Oracle Retail Price Management Installation Guide*.

Installing the RMS Application

For information on steps related to the installation of RMS in secured environment, see Chapter 9 and *Oracle Retail Merchandising System Installation Guide*.

Installing the Allocation Application

For information on steps related to the installation of Allocation in secured environment, see Chapter 20 and *Oracle Retail Allocation Installation Guide*.

5
Troubleshooting

This chapter covers the common errors, issues, and troubleshooting them.

The following topics are covered in this chapter:

Java Version 7 SSL Handshake Issue while Using Self Signed Certificates

Java Version 7 may have issues using self signed certificates. The self-signed root certificate may not be recognized by Java Version 1.7 and a certificate validation exception might be thrown during the SSL handshake. You need to create the private key with Subject Key Identifier to fix this problem. You need to include an option -addext_ski when the orapki utility is used to create the private key in the root wallet.

Importing the Root Certificate in Local Client JRE

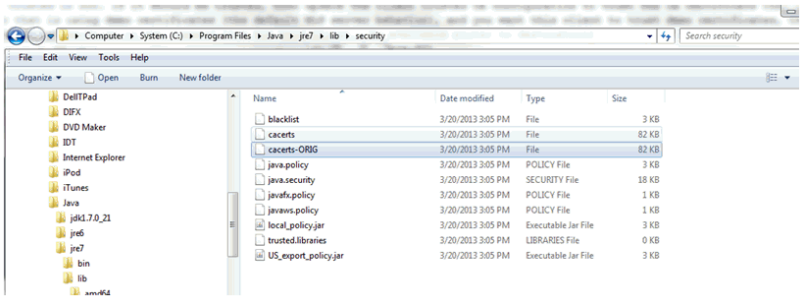
If customers are using certificates other than provided by standard certificate authorities like custom CA implementation, then the JRE used for launching the applications from local machines like laptops or desktops might display a different error messages.

The most probable cause of this issue could be unavailability of root certificates of the CA within the local JRE being used.

Perform the following steps to import the root certificates:

1. Backup cacert at <JRE_HOME>/lib/security/cacert.

Figure 5–1 Cacert Backup



2. Import the certificate using keytool utility as shown in the following example:

```
C:\Program Files\Java\jre7\lib\security>..\..\bin\keytool.exe -import
-trustcacerts -file D:\ADMINISTRATION\SSL\apphost2\Selfsigned\apphost2.root.cer
-alias apphost2 -keystore "C:\Program Files\Java\jre7\lib\security\cacerts"

Enter keystore password: [default is changeit]
Owner: CN=apphost2, OU=<department>, O=<company>,L=<city>,ST=<state or
province>, C=<country>",
Issuer: CN=apphost2, OU=<department>, O=<company>,L=<city>,ST=<state or
province>, C=<country>"
Serial number: 515d4bfb
Valid from: Thu Apr 04 15:16:35 IST 2013 until: Fri Apr 04 15:16:35 IST 2014
Certificate fingerprints:
MD5:  AB:FA:18:2B:BC:FF:1B:67:E7:69:07:2B:DB:E4:C6:D9
SHA1: 2E:98:D4:4B:E0:E7:B6:73:55:4E:5A:BE:C1:9F:EA:9B:71:18:60:BB
SHA256: F3:54:FB:67:80:10:BA:9C:3F:AB:48:0B:27:83:58:BB:3D:22:C5:27:7D:
F4:D1:85:C4:4E:87:57:72:2B:6F:27
Signature algorithm name: SHA1withRSA
Version: 3
Trust this certificate? [no]:  (yes)
Certificate was added to keystore
C:\Program Files\Java\jre7\lib\security>
```

Importing the Root Certificate to the Browser

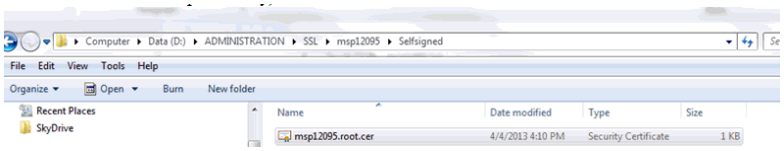
You need to add the signed Weblogic server certificate in the browser to avoid certificate verification error, if the Root Certificate is not in that list of trusted CAs.

Importing the Root Certificate through Internet Explorer

Perform the following steps to import the Root Certificate through Internet Explorer:

1. Copy the Root Certificate file to the workstation.
2. Rename the file to fa_root_cert.cer (this is a quick and easy way to associate the file with the Windows certificate import utility).

Figure 5–2 Importing the Root Certificate File to the Workstation



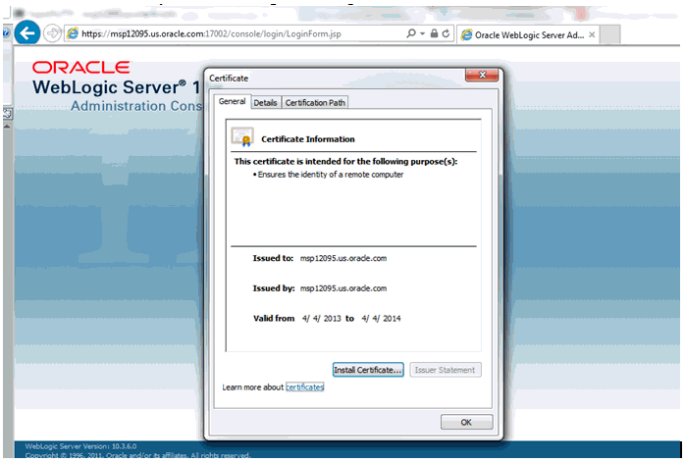
3. Select the file.
4. Click **Install Certificate** and click **Next**.
5. Select **Place all certificates in the following store** and click **Browse**.
6. Select **trusted Root Certification Authorities** and click **OK**.
7. Click **Next**.
8. Click **Finish** and then **Yes** at the Security Warning prompt.
9. Click **OK** to close the remaining open dialog boxes.

Importing the Root Certificate through Mozilla Firefox

Perform the following steps to import the Root Certificate through Mozilla Firefox:

1. Start Mozilla Firefox.
2. Select **Tools** > **Options** from the main menu.
3. Click **Advanced** >**Encryption** tab >**View Certificates**.
4. In Certificate Manager, click the **Authorities** tab and then the **Import** button.
5. In the Downloading Certificate dialog, choose **Trust this CA to identify websites** and click **OK**.
6. Click **OK** in Certificate Manager.
7. Open a browser and test the URL using the SSL port.

Figure 5–3 Importing the Root Certificate File through Mozilla Firefox



Launching Issues with RPM

For launching errors of RPM in the Java console, see the following example:

```
Caused by: java.net.ConnectException: t3s://apphost2:17012: Destination
unreachable; nested exception is:
javax.net.ssl.SSLKeyException: [Security:090542]Certificate chain received from
apphost2 - 10.141.13.195 was not trusted causing SSL handshake failure. Check the
certificate chain to determine if it should be trusted or not. If it should be
trusted, then update the client trusted CA configuration to trust the CA
certificate that signed the peer certificate chain. If you are connecting to a WLS
server that is using demo certificates (the default WLS server behavior), and you
want this client to trust demo certificates, then specify
```

```
-Dweblogic.security.TrustKeyStore=DemoTrust on the command line for this client.;
No available router to destination
at weblogic.rjvm.RJVMFinder.findOrCreateInternal(RJVMFinder.java:216)
at weblogic.rjvm.RJVMFinder.findOrCreate(RJVMFinder.java:170)
at weblogic.rjvm.ServerURL.findOrCreateRJVM(ServerURL.java:153)
at
weblogic.jndi.WLInitialContextFactoryDelegate.getInitialContext(WLInitialContextFa
ctoryDelegate.java:352)
... 27 more
```

The reason could be SSL Handshake failing between Desktop client and the RPM server. Try importing the root certificates in local client JRE (see the steps as provided in Importing the Root Certificate in Local Client JRE section). In case this fails, try disabling hostname verification to NONE for SSL Configuration of the managed server where RPM is deployed. See Disabling Hostname Verification section. This will require restart of the RPM managed server.

Disabling Hostname Verification

The hostname verification ensures that the hostname in the URL to which the client connects matches the hostname in the digital certificate that the server sends back as part of the SSL connection. However, in case SSL handshake is failing due to inability to verify hostname this workaround can be used.

> **Note:** Disabling hostname verification is not recommended on production environments. This is only recommended for testing purposes. Hostname verification helps to prevent man-in-the-middle attacks.

Perform the following steps to disable the hostname verification for testing purposes:

1. Go to **Environment** > **Domain** > **Servers** > **AdminServer**.
2. Click the **SSL** tab.
3. Click **Advanced**.
4. On Hostname Verification, select **NONE**.
5. Save and activate changes.
6. On the Node Manager startup script, look for JAVA. Add the following line:

   Dweblogic.nodemanager.sslHostNameVerificationEnabled=false

   After this change, the script should look as follows:

   ```
   JAVA_OPTIONS="-Dweblogic.nodemanager.sslHostNameVerificationEnabled=false
   ${JAVA_OPTIONS}"
   cd "${NODEMGR_HOME}"
   set -x
   if [ "$LISTEN_PORT" != "" ]
    then
      if [ "$LISTEN_ADDRESS" != "" ]
   ```
7. Restart Node manager.

Verifying the Certificate Content

In situations where the certificate expires or belongs to a different hosts, the certificates become unusable. You can use the keytool utility to determine the details of the

certificate. The certificates should be renewed or new certificates should be obtained from the appropriate certificate authorities, if the certificates expire.

Example:

```
apphost1:[10.3.6_apps] /u00/webadmin/ssl> keytool -printcert -file cert.cer
Certificate[1]:
Owner: CN=apphost1, OU=<department>, O=<company>,L=<city>,ST=<state or province>,
C=<country>"
Issuer: CN=Oracle SSL CA, OU=Class 3 MPKI Secure Server CA, OU=VeriSign Trust
Network, O=Oracle Corporation, C=US
Serial number: 0078dab9f1a5b56e2cd6g92a3987296
Valid from: Thu Oct 11 20:00:00 EDT 2012 until: Sat Oct 12 19:59:59 EDT 2013
Certificate fingerprints:
         MD5:  2B:71:89:11:01:40:43:FC:6F:D7:FB:24:EB:11:A5:1C
         SHA1:
DA:EF:EC:1F:85:A9:DA:0E:E1:1B:50:A6:8B:A8:8A:BA:62:69:35:C1
         SHA256:
C6:6F:6B:A7:C5:2C:9C:3C:40:E3:40:9A:67:18:B9:DC:8A:97:52:DB:FD:AB:4B:E5:B2:56:47:E
C:A7:16:DF:B6
         Signature algorithm name: SHA1withRSA
         Version: 3

Extensions:
```

Verifying the Keystore Content

Keystores are repository of the certificates. If you face issues related to SSL Certificates, you need to check the certificates which are available in the keystore. You need to import the certificates if they are not missing. The keytool command provides the list of the certificates available.

Example:

```
$ keytool -v -list -keystore /u00/webadmin/product/jdk/jre/lib/security/cacerts
$ keytool -v -list -keystore /u00/webadmin/product/10.3.X_APPS/WLS/wlserver_
10.3/server/lib/apphost1.keystore
```

Integration Issues

Oracle Retail applications can be deployed across different hosts and behind network firewalls. Ensure firewalls are configured to allow TCPS connections to enable secure communications among integrated application.

Secured applications using signed certificates need to use same secured protocols for communication. Ensure that all the communicating applications use the same protocol.

For more information on steps to specify secured protocol, see Enforcing Stronger Encryption in WebLogic section.

Communicating applications using signed certificates may need to verify the incoming connections. Root certificates should be available in the keystores of the applications to verify the requests from different host. It is important to import all the root certificates in the keystores of all communicating applications. For information on steps to import the root certificate in local client JRE, see Importing the Root Certificate in Local Client JRE section.

Errors in WLS_FORMS

When you try to restart the WLS_FORMS managed server in Oracle Forms installation after configuring for secure setup (enabling SSL), the managed server startup logs shows the error as shown in Example 5–1. To resolve, ensure that Additional configuration for WLS_FORMS (For forms server) in Pre-installation - Steps for Secured Setup of Oracle Retail Infrastructure in WebLogic have been completed. The startup shows the errors in the logs as shown in the example, when you try to restart the WLS_FORMS managed server in Oracle Forms installation after configuring for security.

Example 5–1 WLS_Forms startup error

```
Feb 6, 2013 6:05:40 AM EST> <Notice> <Cluster> <BEA-000133> <Waiting to
synchronize with other running members of cluster_forms.>
<Feb 6, 2013 6:06:10 AM EST> <Notice> <WebLogicServer> <BEA-000365> <Server state
changed to ADMIN>
<Feb 6, 2013 6:06:10 AM EST> <Notice> <WebLogicServer> <BEA-000365> <Server state
changed to RESUMING>
<Feb 6, 2013 6:06:10 AM EST> <Error> <Cluster> <BEA-003111> <No channel exists for
replication calls for cluster cluster_forms>
<Feb 6, 2013 6:06:10 AM EST> <Critical> <WebLogicServer> <BEA-000386> <Server
subsystem failed. Reason: java.lang.AssertionError: No replication server channel
for WLS_FORMS
java.lang.AssertionError: No replication server channel for WLS_FORMS
        at
weblogic.cluster.replication.ReplicationManagerServerRef.initialize(ReplicationMan
agerServerRef.java:128)
        at
weblogic.cluster.replication.ReplicationManagerServerRef.<clinit>(ReplicationManag
erServerRef.java:84)
        at java.lang.Class.forName0(Native Method)
        at java.lang.Class.forName(Class.java:186)
        at
weblogic.rmi.internal.BasicRuntimeDescriptor.getServerReferenceClass(BasicRuntimeD
escriptor.java:469)
        Truncated. see log file for complete stacktrace
>
<Feb 6, 2013 6:06:10 AM EST> <Notice> <WebLogicServer> <BEA-000365> <Server state
changed to FAILED>
<Feb 6, 2013 6:06:10 AM EST> <Error> <WebLogicServer> <BEA-000383> <A critical
service failed. The server will shut itself down>
<Feb 6, 2013 6:06:10 AM EST> <Notice> <WebLogicServer> <BEA-000365> <Server state
changed to FORCE_SHUTTING_DOWN>
<Feb 6, 2013 6:06:11 AM> <FINEST> <NodeManager> <Waiting for the process to die:
28209>
<Feb 6, 2013 6:06:11 AM> <INFO> <NodeManager> <Server failed during startup so
will not be restarted>
<Feb 6, 2013 6:06:11 AM> <FINEST> <NodeManager> <runMonitor returned, setting
finished=true and notifying waiters>
```

Ensure you have completed the steps mentioned in Additional Configuration for WLS_FORMS (For forms server) section of Chapter 1.

HTTPS Service Encountering Redirect Loop After Applying Policy A

The proxy server access enters into a redirect loop, if the services are secured with policy A (username token over SSL), and the deployment is in a cluster. The access to such services does not work.

Perform the following workaround through SB Console, for services that are secured with HTTPS:

1. Click **Resource Browser**.
2. Click **Proxy Services under Resource Browser**.
3. Click **Create under Change Center** to start a session.
4. For each of the SSL secured proxy services, perform the following steps:
    1. Click the proxy service you want to change.
    2. Click **Edit** next to **HTTP Transport Configuration**.
    3. Uncheck **HTTPS Required** check box.
    4. Click **Last**.
    5. Click **Save**.
5. Click **Activate** and then **Submit**.

6

Importing Topology Certificate

Implementation of SSL into the Oracle Retail deployment is driven by mapping the SSL certificates and wallets to various participating components in the topology.

Importing Certificates into Middleware and Repository of Oracle Retail Applications

Table 6–1 describes the trust stores to be updated while confirming the certificates imported into middleware and repository of Oracle Retail applications. Ensure you have updated the given trust stores with the signed (either self signed or issued by certifying authority) certificates

---

Note: In Table 6–1, the *root.cer are the public key certificates and the *server.cer are the private key certificates.

Table 6–1 Importing Topology Certificate

		Java app-host		Forms app-host		RIB app-host		BIPublisher-host		OID-host	Client-host	
Component	**Certificates**	**Java app -Managed server**	**Java app-JAVA cacerts**	**Forms app - Managed server**	**Forms app-JAVA cacerts**	**RIB app-Managed server**	**RIB app-JAVA cacerts**	**BIPublisher - Managed server**	**BIPublisher - JAVA cacerts**	**Wallet**	**Browser**	**Client-JAVA cacerts**
Java.app	appserver.cer	Yes	No	No	No	No	No	No	No	No	No	No
Java.app	approot.cer	Yes	Yes	No	No	No	Yes	No	Yes	Yes	Yes	Yes
Forms.app	fmserver.cer.	No	No	Yes	No	No	No	No	No	No	No	No
Forms.app	fmroot.cer	No	No	No	Yes	No	No	No	Yes	Yes	Yes	Yes
RIB.app	ribserver.cer	No	No	No	No	Yes	No	No	No	No	No	No
RIB.app	ribroot.cer	No	Yes	No	No	Yes	Yes	No	No	No	Yes	Yes
BI Publisher	biserver.cer	No	No	No	No	No	No	Yes	No	No	No	No
BI Publisher	biroot.cer	No	Yes	No	Yes	No	No	Yes	Yes	No	Yes	Yes
OID	oidcer.cer	No	No	No	No	No	No	No	No	Yes	No	No
OID	oidroot.cer	No	Yes	No	No	No	No	No	Yes	Yes	Yes	Yes

7

Using Self Signed Certificates

Self signed certificates can be used for development environment for securing applications. The generic steps to be followed for creating self signed certificates and configuring for use for Oracle Retail application deployment are covered in the subsequent sections.

The following topics are covered in this chapter:

- Creating a Keystore through the Keytool in Fusion Middleware (FMW) 11g
- Exporting the Certificate from the Identity Keystore into a File
- Importing the Certificate Exported into trust.keystore
- Configuring WebLogic
- Configuring Nodemanager
- Importing Self Signed Root Certificate into Java Virtual Machine (JVM) Trust Store
- Disabling Hostname Verification
- Converting PKCS7 Certificate to x.509 Certificate

Creating a Keystore through the Keytool in Fusion Middleware (FMW) 11g

Perform the following steps to create a keystore through the keytool in Fusion Middleware (FMW) 11g:

1. Create a directory for storing the keystores.

   $ mkdir ssl

2. Run the following to set the environment:

   $ cd $MIDDLEWARE_HOME/user_projects/domains/<domain>/bin

   $ . ./setDomainEnv.sh

   Example:

   ```
   apphost2:[10.3.6_apps] /u00/webadmin/product/10.3.6/WLS/user_
   projects/domains/APPDomain/bin> . ./setDomainEnv.sh
   apphost2:[10.3.6_apps] /u00/webadmin/product/10.3.6/WLS/user_
   projects/domains/APPDomain>
   ```

3. Create a keystore and private key, by executing the following command:

   ```
   keytool -genkey -alias <alias> -keyalg RSA -keysize 2048 -dname <dn> -keypass
   <password> -keystore <keystore> -storepass <password> -validity 365
   ```

   Example:

   ```
   apphost2:[10.3.6_apps] /u00/webadmin/ssl> keytool -genkey -alias apphost2
   ```

```
-keyalg RSA -keysize 2048 -dname "CN=apphost2,OU=RGBU, O=Oracle
Corporation,L=Minneapolis,ST=Minnesota,C=US" -keypass <kpass> -keystore
/u00/webadmin/ssl/apphost2.keystore -storepass <spass> -validity 365

apphost2:[10.3.6_apps] /u00/webadmin/ssl> ls -ltra
total 12
drwxr-xr-x 18 webadmin dba 4096 Apr  4 05:31 ..
-rw-r--r--  1 webadmin dba 2261 Apr  4 05:46 apphost2.keystore
drwxr-xr-x  2 webadmin dba 4096 Apr  4 05:46 .
apphost2:[10.3.6_apps] /u00/webadmin/ssl>
```

Exporting the Certificate from the Identity Keystore into a File

Perform the following steps to export the certificate from the identity keystore into a file (for example, pubkey.cer):

1. Run the following command:

   $ keytool -export -alias selfsignedcert -file pubkey.cer -keystore identity.jks -storepass <password>

   Example:

```
apphost2:[10.3.6_apps] /u00/webadmin/ssl> keytool -export -alias apphost2 -file
/u00/webadmin/ssl/pubkey.cer -keystore /u00/webadmin/ssl/apphost2.keystore
-storepass <spass>
Certificate stored in file </u00/webadmin/ssl/ropubkey.cerot.cer>
apphost2:[10.3.6_apps] /u00/webadmin/ssl> ls -l
total 8
-rw-r--r-- 1 webadmin dba 2261 Apr  4 05:46 apphost2.keystore
-rw-r--r-- 1 webadmin dba  906 Apr  4 06:40 pubkey.cer
apphost2:[10.3.6_apps] /u00/webadmin/ssl>
```

Importing the Certificate Exported into trust.keystore

Perform the following steps to import the certificate you exported into trust.keystore:

1. Run the following command:

   $ keytool -import -alias selfsignedcert -trustcacerts -file pubkey.cer -keystore trust.keystore -storepass <password>

   Example:

```
apphost2:[10.3.6_apps] /u00/webadmin/ssl> keytool -import -alias apphost2
-trustcacerts -file pubkey.cer -keystore trust.keystore -storepass <spass>
Owner: CN=apphost2, OU=RGBU, O=Oracle Corporation, L=Minneapolis, ST=Minnesota,
C=US
Issuer: CN=apphost2, OU=RGBU, O=Oracle Corporation, L=Minneapolis,
ST=Minnesota, C=US
Serial number: 515d4bfb
Valid from: Thu Apr 04 05:46:35 EDT 2013 until: Fri Apr 04 05:46:35 EDT 2014
Certificate fingerprints:
         MD5:  AB:FA:18:2B:BC:FF:1B:67:E7:69:07:2B:DB:E4:C6:D9
         SHA1: 2E:98:D4:4B:E0:E7:B6:73:55:4E:5A:BE:C1:9F:EA:9B:71:18:60:BB
         Signature algorithm name: SHA1withRSA
         Version: 3
Trust this certificate? [no]:  yes
Certificate was added to keystore
apphost2:[10.3.6_apps] /u00/webadmin/ssl>
```

Configuring WebLogic

You need to enable SSL for WebLogic server's Admin and managed servers by following the steps as provided in Configuring the Application Server for SSL section.

Configuring Nodemanager

You need to secure the Node manager by following the steps in Securing Nodemanager with SSL Certificates section.

Importing Self Signed Root Certificate into Java Virtual Machine (JVM) Trust Store

In order for the Java Virtual Machine (JVM) to trust in your newly created certificate, import your custom certificates into your JVM trust store.

Perform the following steps to import the root certificate into JVM Trust Store:

1. Ensure that JAVA_HOME has been already set up.
2. Run the following command:

   $keytool -import -trustcacerts -file rootCer.cer -alias selfsignedcert -keystore cacerts

   Example:

```
apphost2:[10.3.6_apps] /u00/webadmin/product/jdk1.6.0_
30.64bit/jre/lib/security> keytool -import -trustcacerts -file
/u00/webadmin/ssl/root.cer -alias apphost2 -keystore
/u00/webadmin/product/jdk1.6.0_30.64bit/jre/lib/security/cacerts -storepass
[spass default is changeit]
Owner: CN=apphost2, OU=<department>, O=<company>,L=<city>,ST=<state or
province>, C=<country>"
Issuer: CN=apphost2, OU=<department>, O=<company>,L=<city>,ST=<state or
province>, C=<country>"
Serial number: 515d4bfb
Valid from: Thu Apr 04 05:46:35 EDT 2013 until: Fri Apr 04 05:46:35 EDT 2014
Certificate fingerprints:
         MD5:  AB:FA:18:2B:BC:FF:1B:67:E7:69:07:2B:DB:E4:C6:D9
         SHA1: 2E:98:D4:4B:E0:E7:B6:73:55:4E:5A:BE:C1:9F:EA:9B:71:18:60:BB
         Signature algorithm name: SHA1withRSA
         Version: 3
Trust this certificate? [no]:  yes
Certificate was added to keystore
apphost2:[10.3.6_apps] /u00/webadmin/product/jdk1.6.0_
30.64bit/jre/lib/security>
```

Disabling Hostname Verification

This section has been covered under Disabling Hostname Verification section.

Converting PKCS7 Certificate to x.509 Certificate

Certificate authorities provide signed certificates of different formats. However, not all formats of certificates can be imported to Java based keystores. Hence the certificates need to be converted to usable form. Java based Keystores supports x.509 format of certificate.

The following example demonstrates converting certificate PKCS 7 to x.509 format:

1. Copy the PKCS 7 certificate file to a Windows desktop.
2. Rename the file and provide .p7b extension.
3. Open the .p7b file.
4. Click the plus (+) symbol.
5. Click the Certificates directory.

   An Intermediary certificate if provided by CA for trust.

   Note: If an Extended Validation certificate is being converted you should see three files. The End Entity certificate and the two EV intermediate CA's.

6. Right click on your certificate file.
7. Select All Tasks > Export.
8. Click **Next**.
9. Select Base-64 encoded X.509 (.cer) > click Next.
10. Browse to a location to store the file.
11. Enter a File name.

    For example, MyCert. The .cer extension is added automatically.
12. Click **Save**.
13. Click **Next**.
14. Click **Save**.

The certificate can be now imported into java based keystores.

Example:

```
apphost1:[10.3.6_apps] /u00/webadmin/ssl> keytool -import -trustcacerts -alias
apphost1 -file /u00/webadmin/ssl/cert-x509.cer -keystore
/u00/webadmin/product/jdk/jre/lib/security/cacerts
Enter keystore password: [default is changeit]
Certificate was added to keystore
apphost1:[10.3.6_apps] /u00/webadmin/ssl>
```

Part II

Oracle Retail Merchandising System (RMS)

The following chapters provide guidance for administrators, developers, and system integrators who securely administer, customize, and integrate the Oracle Retail Merchandising System (RMS) application.

Part II contains the following chapters:

8

Understanding Security

This chapter covers the technical overview of the authentication process used for RMS, Oracle Retail Trade Management (RTM), and Oracle Retail Sales Audit (ReSA) modules using Oracle Access manager and Single Sign-On.

Further, it details the security considerations and implementations that are part of the Database, Data level security, as well as the Application layer. It also covers encryption and hashing techniques used to secure credit card information received in ReSA.

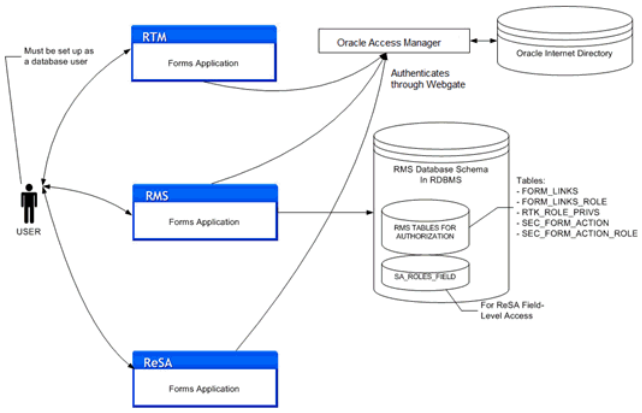
The following topics are covered in this chapter:

- Technical Overview of the Security Features
- Single Sign-On (SSO) for Oracle Retail Forms Application
- Security Features of the Application
- Encryption and Hashing

Technical Overview of the Security Features

Figure 8–1 shows the security model for RMS, RTM, and ReSA Applications.

Figure 8–1 Security Model for RMS, RTM, and ReSA Applications



Single Sign-On (SSO) for Oracle Retail Forms Application

The logical component diagram depicts the typical Oracle Forms Application and its use of the Single Sign-On (SSO) features within the Oracle Forms framework. The

Oracle Internet Directory (OID) user store acts as an access control mechanism to the Oracle Forms Application when its Web-interface is accessed through the Workspace.

Figure 8–2 Logical Component of SSO for Oracle Retail Forms Application

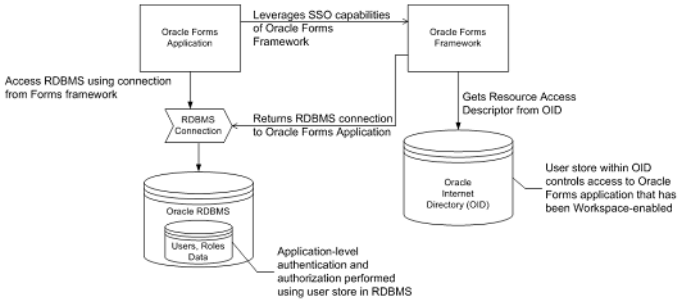


Figure 8–2 outlines the interaction between the Oracle Forms Framework, OID, and the actual Forms Application.

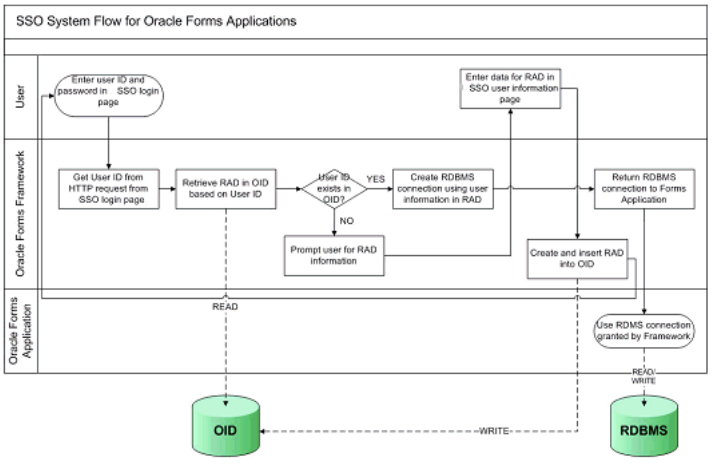
The Forms Application is deployed as a Web Application in Oracle Application Server (OAS); the Forms Application is SSO-capable. You need to modify the formsweb.cfg to enable SSO support.

The formsweb.cfg file also includes the following two lines to depict the SSOMode through which OAM authenticates OID:

ssoMode=webgate

ssoDynamicResourceCreate=true

Figure 8–3 SSO System Flow for Oracle Form Applications



Note: ReSA and RTM share the same database as that of RMS. Security features that are applicable to RMS are applicable to ReSA and RTM as well.

Security Features of the Application

The security features of the Application are as follows:

- **Access Control** - It is the process of restricting access to a particular entity based upon a broad range of criteria that may or may not include the attributes related to a particular user.
- **Authentication** - It is the process of verifying the identity of a user. The authentication process usually requires a user to provide a user name and password or a combination thereof, upon signing into an application.
- **Authorization** - It is the process of checking to see if an authenticated user has the privilege to access particular system functionality.
- **Data Authorization** - It is the process of determining an authenticated user's rights to act upon a particular set of data. This process typically checks if the authenticated user is linked to a certain level in the organization hierarchy and/or a certain level in the merchandise hierarchy.
- **Role-Based Access** - Within the Oracle Retail's systems, users are assigned to different roles. The role logical grouping has different access rights to specific functions within the various Oracle Retail Systems.
- **User Attributes** - It is the data that is associated with a particular user, and may be used to define a particular user. User attributes do not impact authentication, authorization, or data authorization.
- **User Store** - It is a repository that holds user data required for authentication and authorization processes.

Data authorization in RMS allows definition of who can view and therefore perform the actions associated with a role, on a given data. Data authorization is implemented throughout RMS using security groups. As necessary, security groups are created and associated with levels of the merchandise hierarchy or levels of the organizational hierarchy.

Security groups are powerful tools for data authorization in RMS; however, they require significant administration. Security groups are defined on RMS tables, so it is possible that the information can be interfaced onto these tables from an external system aware of the merchandise and organizational hierarchies.

There are essentially two major types of item/location security in RMS: Find Form and LOVs/validation. Both of these methods rely on relationships defined between groups of users and merchandise and organizational data.

Security groups are defined on the SEC_GROUP table. These security groups are meant to define users with many job tasks who need to access the same information.

SEC_GROUP

The SEC_GROUP table stores group attributes. Security groups allow users who need similar data to be grouped together.

- **GROUP_ID** - This contains the unique identifier associated with the security group.
- **GROUP_NAME** - This contains the name of the security group.
- **ROLE** - This contains the role that a client wants to assign to this group. This field is referenced in the code type ROLE. There are no pre-defined values for this field and it is completely user-defined. This field does not have any functionality linking it to Oracle Roles or any other type of roles used within the RMS. This field is used within the regionality dialog for searching and reporting.

Security groups are defined in the RMS user interface. Users are added as members of a group by associations on the SEC_USER_GROUP table.

SEC_USER_GROUP

The SEC_USER_GROUP table link users.

The security groups are as follows:

- **GROUP ID** - This contains the unique identifier associated with the security group as defined in the SEC_GROUP table.
- **USER ID** - This contains the database/application username of user associated with the current security group

The security groups can be associated with specific locations and zones using the SEC_GROUP_ZONE_MATRIX and SEC_GROUP_LOC_MATRIX tables.

The security group information is also used to determine the information that users have access to in RMS Find Forms and LOVs/field validation. If a user is associated with a security group that has access to a limited range of items and the user searches in the item Find Forms, the search will only return results that are in the items the user has access to. If the same user enters an item number in an item field, validation will ensure that the user has access to this item. Any item LOVs will also display only the user's list of items.

The intersection between security user groups and merchandise hierarchy levels are stored on the FILTER_MERCH_HIER table.

- **SEC_GROUP_ID** - This is the ID of the User Security group as defined on the SEC_GROUP table.
- **FILTER_MERCH_LEVEL** - This is the merchandise hierarchy level assigned to the User Security Group. This can be a code representing group, department, class, or subclass.
- **FILTER_MERCH_ID** - This is a group or department included in the filtering.
- **FILTER_MERCH_ID_CLASS** - This is the class under the department which is included in the filtering.
- **FILTER_MERCH_ID_SUBCLASS** - This is the subclass under the class which is included in the filtering.

The intersection between security user groups and organizational hierarchy level is stored in FILTER_ORG_HIER table.

- **SEC_GROUP_ID** - This is the ID of the User Security group as defined on the SEC_GROUP table.
- **FILTER_ORG_LEVEL** - This is the Organization hierarchy level assigned to the User Security Group. Valid values are contained in the CODE_DETAIL table with a CODE_TYPE of FLOW.
- **FILTER_ORG_ID** - This is the ID of the Organization hierarchy level assigned to the User Security Group.

RMS includes two install scripts that sets up some of this security information. The install script superGroup.sql creates a security group that must not to be associated with any level of the merchandise or organizational hierarchy. Members of this group will have access to all data in the Application. The install script superUser.sql associates the user running the script with the superUser group, and therefore ensures that the user running the script will have access to all data in the system.

RMS Users and Security

You need to create user roles within RMS and provide users with a mechanism for accessing the system. When security is leveraged, it controls a user's access to individual application functions and data sets.

Users in RMS are set up by the database administrator using a user creation script. RMS users are database users that connect directly to the database to access RMS. Each database user has synonyms to the master RMS schema and is able to update and modify data within that schema, and cannot change the structure of the tables and objects. This user structure is specific to RMS, ReSA, and RTM. Applications such as RPM, Allocation, and ReIM use other means to manage users. User management for each application is discussed within its respective section.

In order to ensure users are limited to parts of the application and information that is relevant to their business role, RMS has a three-tier security structure.

The three levels of security offered by RMS are as follows:

- **Database-level security** - This is a built in feature of Oracle Database, based on database roles.
- **Application-level security** - This is a form or screen-level security based on database roles.
- **Data-level security** - This is built into RMS to give a client the ability to further limit user access to information.

Database-level security

For information on this section, see Chapter 2.

Application-level security

The application-level security restricts the user's access to either entire areas of the system (for example, Purchase Orders) or restricts the modes in which users can access areas (for example, viewing Purchase Orders only).

The application level security covers the following five primary components:

- **Menu level security** - This allows a client to determine which menu options are available to a user based on that user's Oracle Role. Menu level security is set up using the SEC_FORM_ACTION and SEC_FORM_ACTION_ROLE tables to define what options each Oracle Role has access to for the various menus in RMS.
- **Navigator** - When RMS is launched, the user operates from a folder driven interface on the Oracle Retail Start Form. These folders provide access to all of the functional areas available in RMS. Navigator security is established through the Start Tree Administration Form. From here, the client defines the folder tree structure in the system in addition to defining security access to forms throughout the tree by Oracle Role. A client can also develop a script to perform these updates. The updates affect the NAV_FOLDER, NAV_ELEMENT, NAV_ELEMENT_MODE, and NAV_ELEMENT_MODE_ROLE tables. Like the form (which updates these tables), these tables control the folder tree structure, the forms accessible from the tree structure, and the user roles cleared for access to the different forms.

  The following are some examples around NAV table customization:

  – The User roles must be associated with main menu objects. These associations are defined in NAV_ELEMENT_MODE_ROLE table. The Functional role in the organization will determine which areas of the application you can use.

For example, a buyer would have access to most areas of the application whereas a junior level employee with limited database roles might not access to financial areas.

  - The association between the main menu options and roles can be built by RMS install script navrole.sql. This script prompts for a role and associates that role with all main menu options. Based on the security requirement of the client, you may choose to alter NAV_ELEMENT_MODE_ROLE table to restrict access to all main menu options to all users. Basically, after running the navrole.sql script as-is, the client can remove the rows from the table to ensure restricted access to users.
  - A client can also customize the application to include additional options from the main menu. In such cases, the client should also create a customized script to add this information to NAV_ELEMENT_MODE_ROLE table. Customized scripts must be preserved for future upgrades and patches.

- **Find Forms** - This is the most common entry point into RMS core functionality. Within this form the user typically has the option to select what type of action they want to perform (New, Edit, or View) from an action list box. Find Forms security is set up through the P_SECURITY package within the Find Forms. This logic is hard-coded into each of the Find Forms. As a result, these packages have to be modified once Oracle Roles have been defined for a client if customized security is required.
- **Hierarchy Forms** - This Form differs from Find Forms. Find Forms have New, Edit, and View buttons. Hierarchy Form security is managed through the P_SECURITY package within the forms. However, it differs from the Find Forms. The Hierarchy Form leverages the SEC_FORM_ACTION and SEC_FORM_ACTION_ROLE tables to drive what options each Oracle Role has access to.
- **Form Link Security** - The FORM_LINK table allows a client to restrict access and visibility of certain item maintenance sub forms by Oracle Role.

Data-level security

Data-level security restricts user access to specific data within the merchandising system. The client has the ability to limit user data access both from a merchandise hierarchy perspective in addition to an organizational hierarchy perspective. For example, a buyer for the Small Appliances department could have data level security put in place so that they only have the ability to access items within the Small Appliances department. This prevents users from accessing information that does not pertain to their job.

Unlike the other layers of security, this level of security can be configured by the client in the RMS application as follows:

1. Define groups within the organization and merchandise hierarchy and group hierarchy information for these groups.
2. Use the Security Group Maintenance Form to establish each of the groups defined in Step 1 in RMS.
3. Use the Filter Level Group Hierarchy Maintenance Form to establish the relationships between the groups built in Step 2 and the merchandise and organizational hierarchy components are cleared to access based on the analysis of Step 1.
4. Use the Security User/Group Link Form to establish the relationships between the groups defined in Step 2 and the users tied to those groups (as established in Step 1).

The Users that are not associated to a security group or to a security group that was not associated to any parts of the organizational or merchandise hierarchy are considered Super Users in terms of data. They have access to all data within the system.

When all of the security tools for RMS are effectively leveraged by a client a user can become a powerful tool to not only grant access to a system, but to ensure that every employee is systematically focused on the aspect of the client's business that they are responsible for.

Encryption and Hashing

Encryption and hashing techniques are a part of Oracle Advanced Security, hence System and Database Administrators are recommended to refer *Oracle Database Advanced Security Administrator's Guide 11g Release 2* for more details.

The details of encryption and hashing are as follows:

- ORDCUST and SVC_FULFILORDCUST tables are in encrypted tablespace as it stores sensitive information about the customers
- The credit card information that is received for ReSA is encrypted. Sacrypt.pc batch program decrypts the information using DBMS_CRYPTO
  - The program obtains the raw data, converts the encrypted data in text format, pads the obtained raw data with the input encrypted chunk, and decrypts it by calling DBMS_CRYPTO.DECRYPT

    The following items are passed in as input parameters:

    * **Input Chunk Raw** - The padded encrypted raw data for hex 31, with the input chunk to generalize the end of decrypted data for all input chunks
    * **Encryption Type** - The value passed in is - DBMS_CRYPTO.ENCRYP_AES128 + DBMS_CRYPTO.CHAIN_ECB + DBMS_CRYPTO.PAD_PKCS5
    * **Raw Key** - The value passed in is - UTL_ENCODE.BASE64_DECODE(UTIL_RAW.CAST_TO_RAW (Key_Value)). The Key_Value is retrieved from the key file passed in
- RETAIL_SERVICE_REPORT_URL table is used to hold the retail service code, retail service name, and URL for the Web services for Oracle Retail Financial Integration (RFI). The column sys_account is an encrypted column which stores the system account name.

9

Post Installation - Application Administration

This chapter covers the administration tasks performed during post installation of RMS application. The section covers the roles and permissions granted for Oracle Advanced Queuing, setting up table level auditing through the application UI, Security Policies around views, formweb.cfg, and the steps to secure the Webservice calls.

It further covers important security aspects of Data Access Schema (DAS), a new component built in response to specific F release Customer Order Requirements using Oracle Database Replication.

This chapter covers the following topics:

Roles and Permissions

Some online processes are moved to asynchronous processing in RMS UI. Some processes that take a longer time have been moved to background asynchronous processing. This is done by using Oracle's Advanced Queuing (AQ) to push long-running processes of a workflow into a separate and asynchronous transaction. One feature of the async processing is the Async Notification message. This mechanism informs the user when the async processing has been completed. This notification message is displayed to the user in RMS UI in near real time.

The following new AQ objects are introduced to implement the roles and permissions:

- New User RMS_ASYNC_USER is created with _ADMINISTRATOR_ROLE, DBMS_AQ (execute) and DBMS_AQADM (execute) permissions only
  - This new user is a dedicated user added to the RMS install. The RMS_NOTIFICATION_QUEUE queue is also created in this user instead of the owning schema

Views

In RMS application (and also ReSA) as part of security, all search screens and LOVs (List of Values) access the views. Those users who have privileges will only be able to see the data from these views.

These views are implemented by the following scripts:

1. add_filter_policy.sql
   - This script adds filter policy to four main categories of data in RMS
     - Organizational Hierarchy Filtering Policies: Views under this category are V_CHAIN, V_AREA, V_REGION, V_DISTRICT, V_STORE, V_WH, V_EXTERNAL_FINISHER, V_INTERNAL_FINISHER and V_TSF_ENTITY
     - Merchandise Hierarchy Filtering Policies: V_DIVISION, V_GROUPS, V_DEPS, V_CLASS, V_SUBCLASS and V_ITEM_MASTER
     - Data Element Filtering Policies: V_DIFF_GROUP_HEAD, V_LOC_LIST_HEAD, V_LOC_TRAITS, V_SEASONS, V_SKULIST_HEAD, V_TICKET_TYPE_HEAD, V_UDA and V_SUPS
     - Product Location Security Policies: V_TRANSFER_FROM_STORE, V_TRANSFER_FROM_WH, V_TRANSFER_TO_STORE and V_TRANSFER_TO_WH
2. add_cc_sec_policy.sql
   - This script applies credit card security policy to ReSA tables which includes SA_TRAN_TENDER and SA_TRAN_TENDER_REV

Other Common Application Administration

This section covers the common Application Administration.

Data Access Schema (DAS) - Overview

Data Access Schema (DAS) is an extension of RMS database used by external applications requiring RMS data. The external applications extracting data from RMS increases the server load on the main RMS schema which impacts the performance of core RMS functionality. The data in the DAS schema is replicated real time from the main RMS schema allowing the server load of the extract job to be restricted to this separate database.

The core RMS tables are replicated to the DAS schema in near real time. On top of these replicated tables, there is a layer of data base views which structures the data and exposes them as business data. An external application that is not aware of the RMS data structure can read the business data exposed.

For a new external application required data extract from RMS, the client should check if the existing tables and views in the DAS scheme is sufficient. If not, the RMS table should be included in the list of replicated tables. The client should also build additional views on top of the table or views to structure the data in way it is useful to the requesting application while also considering the reuse for the view for any future use.

The DAS schema is exposed to the external application as a read only schema and external applications should not write into this schema:

- Database tables which are replicated from main RMS schema should not be accessed directly from external application. The external application should always access the data using the views only. This will avoid changes to the external application when there is a change in RMS table structure. Any change in RMS data structure should be transparent to the external application just by changing the view built on these tables.

- The external application should not be granted on the database views. The access to the views should be restricted based on the functional data requirement for a specific application.

Application Specific Feature Administration

Following is the application specific feature administration:

- The user can try logging into the application for a specific number of attempts. After 'X' times of continuous failure, the user cannot log in to the application as the account will get locked. For more information, see Configuring the Database Password Policies section.
- Application server session time-out. For more information, see RMS - Forms Timeout Setting - Optional section.
- When the RMS application is accessed without any config parameters and other parameters, default config section should be modified to throw an error message or a blank form. To enforce this security, add the parameter restrictedURLchars=userid to formsweb.cfg.

Example 9–1 formsweb.cfg entry for restricted URL

```
<MW_HOME>/user_projects/domains/ClassicDomain/config/fmwconfig/servers/WLS_
FORMS/applications/formsapp_11.1.2/config/formsweb.cfg

[rmsFqa3]
envfile=./develop/rmsFqa3.env
restrictedURLchars=userid
width=950
height=685
separateFrame=true
form=rtkstrt.fmx
lookAndFeel=Oracle
colorScheme=swan
archive=frmall.jar,icons.jar
imageBase=codebase
heartbeat=12
```

Example - RMS Applications Audit Llog

Audit Trails - The AUDIT_FLD and AUDIT_TBL tables in RMS hold the master table (RMS table names) and field names that are to be audited.

The following are the details for RMS application audit logs:

- The batch auditsys.pc adds audit logic to the tables present in AUDIT_TBL table.
- A table that has an audit requested raised against it has an audit table created to hold all the inserts/updates/deletes of data.
- When this batch program is run, it creates a trigger RMS_tablename_AU which needs to be applied to the master table (tablename). This batch also creates a table tablename_AU.
- This audit table (named [RMS_table]_AU) holds the key values of the master table and the username and date of the audited transaction.
- Additional fields to be audited may be added or removed at the user's request prior to running auditsys.pc program.

- You need to run the auditprg.pc program once the audit table is created to remove the audit table and then add to the RMS audit trail again for any changes to the columns being tracked.

The audit table and the database trigger are automatically promoted to the database.

A user that is granted the following special privileges or a user that has database administrator (DBA) privileges must execute this program:

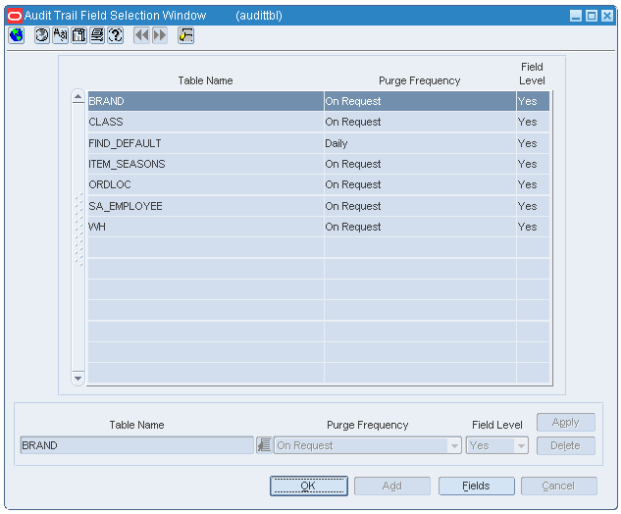
- Create any table
- Create any trigger

See the following example:

You need to make an entry in AUDIT_TBL and AUDIT_FLD tables to enable audit on SA_EMPLOYEE table.

Navigate to **Control** > **System** >**Audit Trail**.

Figure 9–1 Audit Trail Field Selection Window



Fields of SA_EMPLOYEE table to be audited are entered in AUDIT_FLD table. When auditsys.pc program runs, SA_EMPLOYEE_AU table with the columns mentioned in AUDIT_FLD table is created. For each field mentioned in AUDIT_FLD table, OLD_ and NEW_ columns are created by auditsys.pc program. A corresponding trigger is created for the table to capture the audit (old and new changes). In the above example, trigger RMS_SA_EMPLOYEE_AU is created to capture the changes of the mentioned fields.

Post Installation Steps for Webservice Security

You need to configure the user credentials and other security related information at the service consumer and the app service provider layers, in order to provide end to end security between web service consumer and the provider.

Note:

1. The following steps are used for webservices deployed according to the Oracle Retail Installation guides.
2. The following steps are not applicable for RSB. For more information, see the *Oracle Retail Service Backbone Security Guide*.

Applying Policy A

Applying policy A involves the following:

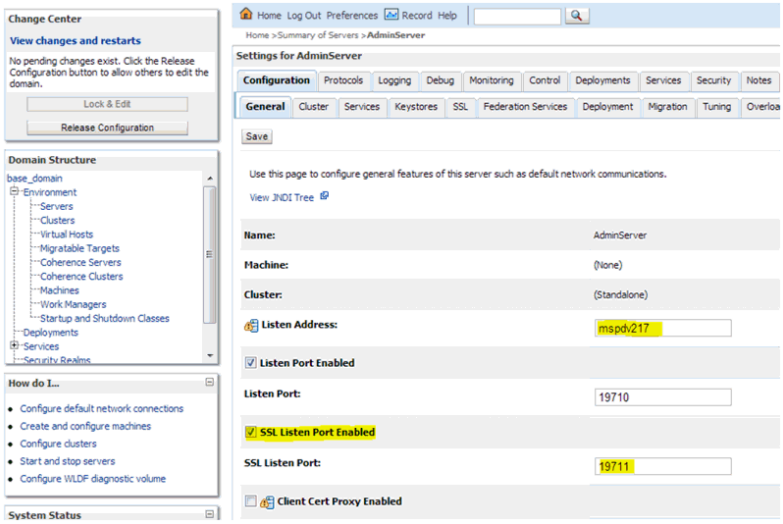
- Enabling the HTTPS servers
- Creating the Webservice users
- Securing services
- Updating the Webservice deployment
- Webservice Clock Skew setting

Enabling the HTTPS servers

Perform the following steps to enable HTTPS servers:

1. In WebLogic Admin Console, click Environment > Servers.
2. Click the server where the web service has been deployed.
3. Click the **General** tab.
4. Check the SSL Listen Port Enabled check box.
5. Enter a port number for the SSL Listen Port. This is the port number for service end point.
6. Enter the hostname in Listen Address field.
7. Click **Save**.

Figure 9–2 Enabling the HTTPS Servers



Creating the Webservice User

Perform the following steps to create roles and users who can access the Web services:

1. In WebLogic Admin Console, click the Domain Structure window, and click the Security Realms link.

   The default realm appears.

2. Click the link on the realm.
3. Click the **Users and Groups** tab.
4. Click **New**.
5. Enter the user name and password details on the next screen.
6. Leave the default value for Provider.
7. Click **OK** to save the changes.

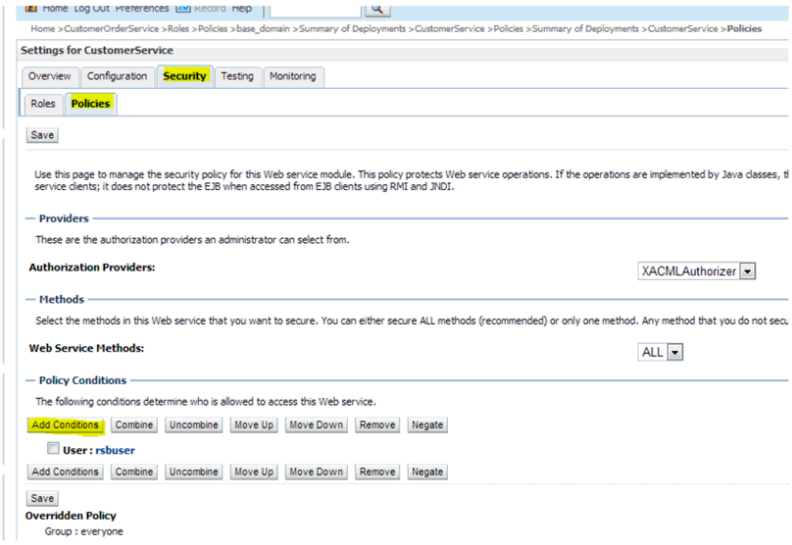
   The new user is shown in the list of users.

Securing services

Perform the following steps in WebLogic Admin Console for each of the services to be secured:

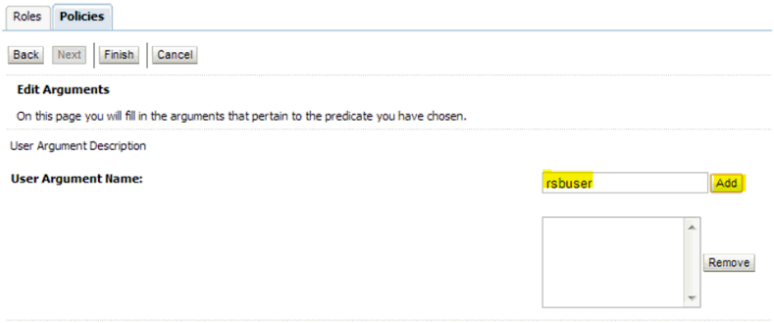
1. Attach the user created in previous step to the service.
2. Click Deployments.
3. Click the service you want to secure.
4. Click **Securities** and then **Policies**.

Figure 9–3 Securing Services



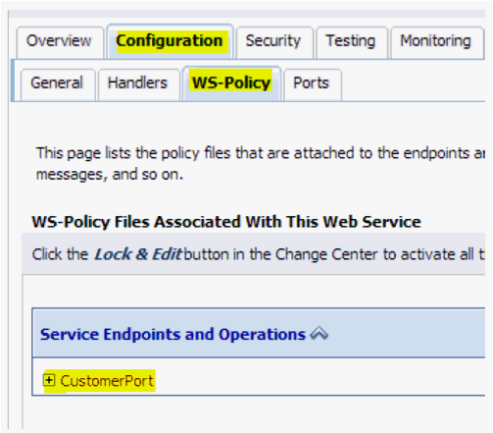
5. Click Add **Conditions** > Predict List: Pick User from dropdown > Next > User Argument Name: > Type username you created > Add > Finish > Save.

Figure 9–4 Add Conditions Window



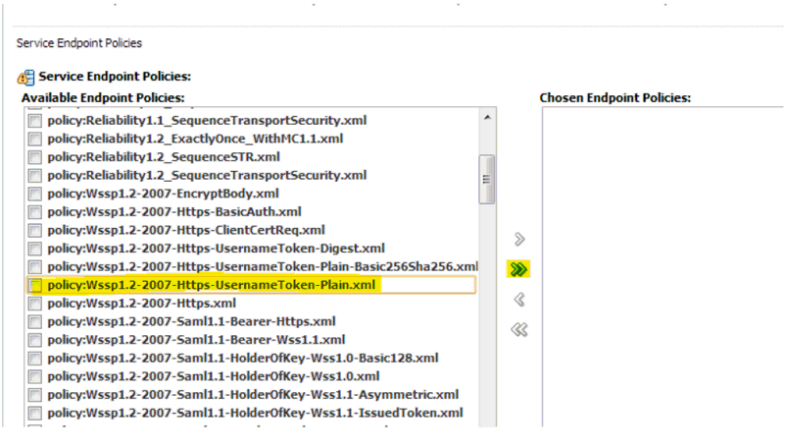
6. Attach policy to the service.
7. Navigate to **Configuration** tab.
8. Click **WSB Policy** tab and select the service port.

Figure 9–5 Attaching WS Policy to the Service



9. Pick WebLogic > Next > Service Endpoint Policies: select policy:Wssp1.22007HttpsUsernameTokenPlain.xml > Finish

Figure 9–6 Service Endpoint Policies



10. Click **OK** if WebLogic prompts you to save Plan.xml.

Updating the Webservice deployment

Perform the following steps to update the Webservice deployment:

1. In WebLogic Admin Console, click **Deployments**.
2. Click **Lock & Edit** and select the deployed application which has the Webservices to be secured.
3. Click **Update** and select the deployment ear along with the Plan.xml if saved in the previous steps.
4. Click **Finish**.
5. Click **Activate Changes** to reflect the changes.
6. Verify the configuration by checking the WSDL of the service.

   The WSDL must have the policy information in it.

Webservice Clock Skew setting

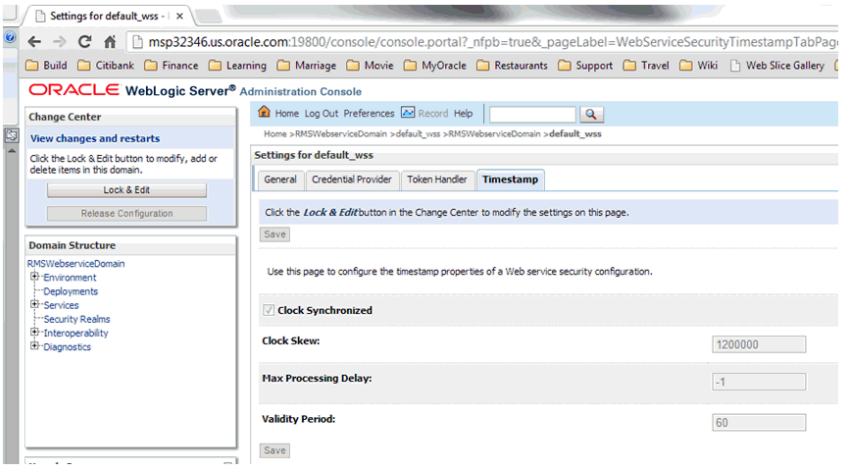
Webservices when secured need to be time synched with providers and consumers. However, due to various reasons the providers and consumers can have different time gap.

Weblogic can be configured to different tolerance level for webservices to work.

Perform the following steps to set the tolerance level of time different:

1. Navigate to WLS Console > Domain > Web Service Security > default_wss > Timestamp.
2. Click **Lock and Edit** and update the **Clock Skew** with tolerance limit (in milliseconds).
3. Click **Activate Changes**.

Figure 9–7 Setting the Tolerance Level of Time Different



4. Bounce the managed server hosting Webservice once the changes are implemented.

Applying Policy B

Applying policy B involves the following:

- Creating the Webservice users
- Securing services
- Updating the Webservice deployment

Creating the Webservice user

Perform the following steps to create roles and users who can access the Web services:

1. In WebLogic Admin Console, click the Domain Structure window, and click the Security Realms link.

   The default realm appears.
2. Click the link on the realm.
3. Click the **Users and Groups** tab.
4. Click **New**.
5. Enter the user name and password details on the next screen.
6. Leave the default value for Provider.
7. Click **OK** to save the changes.

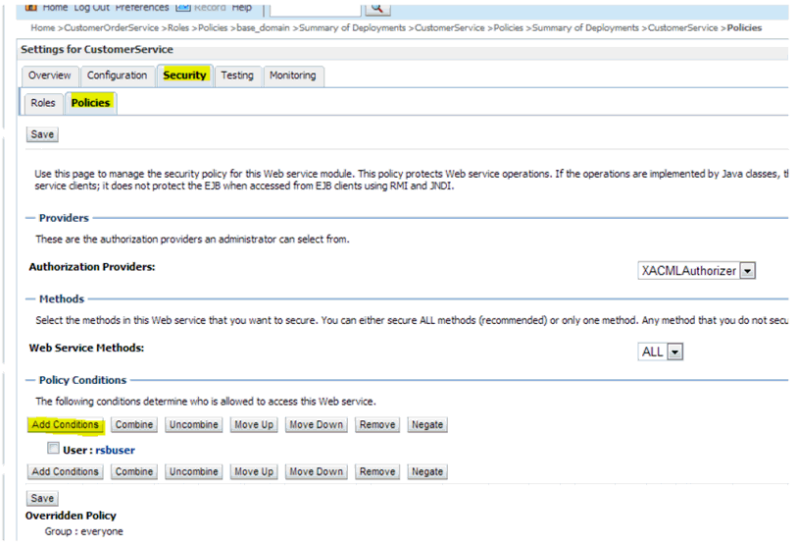
   The new user is shown in the list of users.

Securing services

Perform the following steps in WebLogic Admin Console for each of the services to be secured:

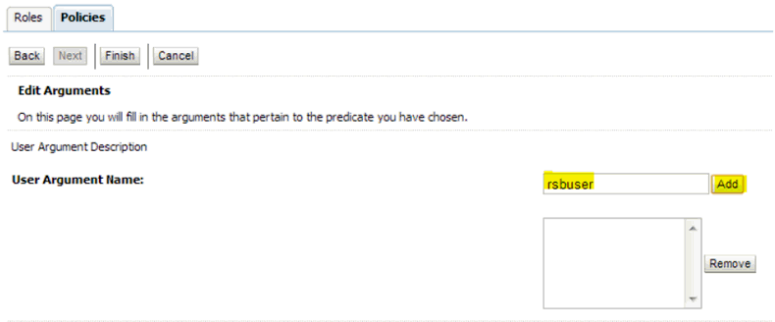
1. Attach the user created in previous step to the service.
2. Click Deployments.
3. Click the service you want to secure.
4. Click **Securities** and then **Policies**.

Figure 9–8 Securing Services



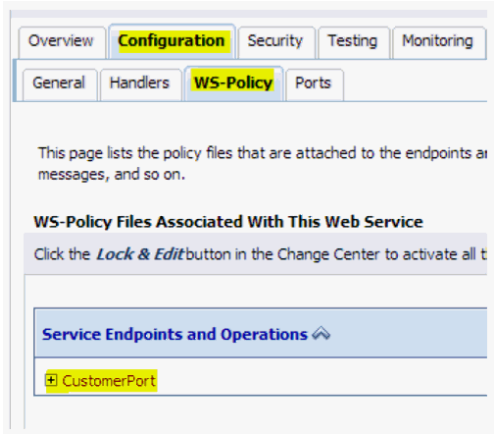
5. Click Add **Conditions** > Predict List: Pick User from dropdown > Next > User Argument Name: > Type username you created > Add > Finish > Save.

Figure 9–9 Add Conditions Window



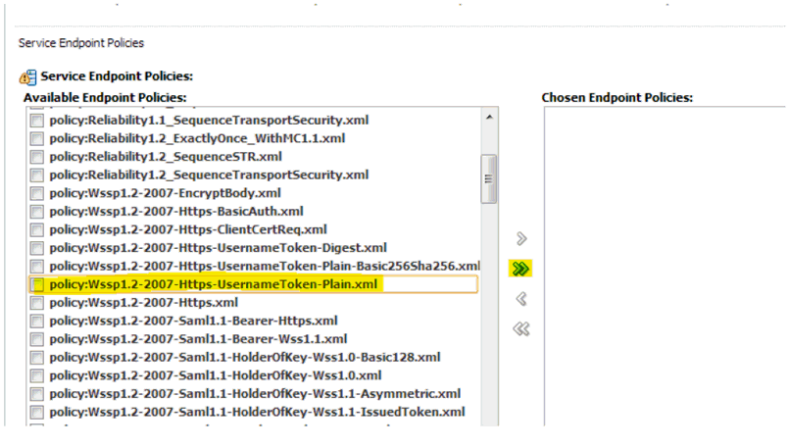
6. Attach policy to the service.
7. Navigate to **Configuration** tab.
8. Click **WSB Policy** tab and select the service port.

Figure 9–10 Attaching WS Policy to the Service



9. Pick WebLogic > Next > Service Endpoint Policies: select policy:Wssp1.22007HttpsUsernameTokenPlain.xml > Finish

Figure 9–11 Service Endpoint Policies



10. Click **OK** if WebLogic prompts you to save Plan.xml.

Updating the Webservice deployment

Perform the following steps to update the Webservice deployment:

1. In WebLogic Admin Console, click **Deployments**.
2. Click **Lock & Edit** and select the deployed application which has the Webservices to be secured.
3. Click **Update** and select the deployment ear along with the Plan.xml if saved in the previous steps.
4. Click **Finish**.
5. Click **Activate Changes** to reflect the changes.
6. Verify the configuration by checking the WSDL of the service.

   The WSDL must have the policy information in it.

Part III

Oracle Retail Invoice Matching (ReIM)

The following chapters provide guidance for administrators, developers, and system integrators who securely administer, customize, and integrate the Oracle Retail Invoice Matching (ReIM) application.

Part III contains the following chapters:

10

General Security Considerations

This chapter discusses how to securely install the Oracle Retail Invoice Matching (ReIM) application. To obtain a secure configuration, follow the instructions and advice provided below.

The ReIM application is installed on the server, but is used in the distributed environment. Both client and server security should be taken into consideration when hardening application deployment. You need to reference your desktop and server operation system security guides, if available for more information on reinforcing security for the execution environment.

In particular, only valid users should have access to the client workstations running clients for the application. The reasonable locking policy should be established to lock out computer screens after some time of inactivity. The Security policy should be established at the desktop level to monitor unsuccessful login attempts. The System administrator should guarantee that the operation system has the latest mandatory update patches.

You should use only the supported browsers to access the application. The browsers should be patched with all the mandatory security updates dictated by the browser's vendor. Browser auto-complete feature should be disabled by the system administrator. It is advised to add the server ReIM is deployed at to the list of sites for Local Intranet in the Web browser. The browser should be allowed to open pop-up initiated by the Invoice Matching application. Also the browser should allow submitting non-encrypted form data.

For more information on how to secure the internal network used to access Invoice Matching server(s), see the *Network Security Configuration Guide*. This should include both physical and logical security of the network. Only SSL enabled communication should be used.

Only the desktops on the intranet should be allowed accessing the application server. The best approach is to limit the set of client computers on the network that can access the application server. That can be done at the network level to prevent guest users on the local network from even seeing the application server.

Only system administrators should have access to the application server(s). Business users (even power users) should not have accounts on the application server computers. If such accounts do exist, the OS account privileges should prevent business user from accessing application server files/directories associated with Invoice Matching application.

The users running batches should never be system administrators. The best approach is to have a single dedicated user for running batches, rather than having multiple users running batches ad hoc. At the same time Invoice Matching application does not prevent any valid application user from running batches.

Only authorized users should be able to upload/download files consumed/produced by Invoice Matching application. The directory structure for incoming EDI files should be accessible to the OS user running batches as read only. It is recommended to keep outbound files in a separate directory. The outbound files directory should allow write access. Also it is recommended to have separate write accessible directories for rejected files and for audit files.

It is recommended to keep audit copies of the processed files. It is up to the retailer to provide the process for that. Audit copies should be created prior to supplying the files for ReIM processing. The files can be kept in an audit directory or using any other appropriate document management system that would allow easy retrieval later on.

Note: ReIM processes files that have been supplied by vendors or that have been supplied by RMS. Secure file transfer should be deployed in both cases.

11

Understanding Security

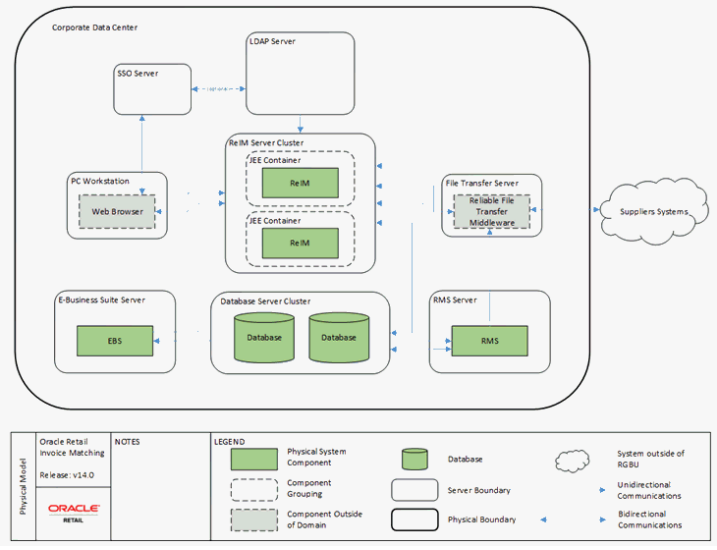
This chapter provides an overview of all the security features available in the ReIM application. It is provided as a reference for how the application securely communicates with other applications; how the application authenticates, authorizes, and audits users; and the encryption and hashing mechanisms used by the application.

The following topics are covered in this chapter:

Security Features Overview

Figure 11–1 shows the physical deployment of ReIM.

Figure 11–1 ReIM Physical Deployment



The Application Server is deployed on the corporate intranet on WebLogic cluster. The Application Server is connected to a cluster of Oracle databases and a corporate LDAP server. Optionally Single Sign-On infrastructure can be deployed. In addition secure FTP server is responsible for transferring files in/out of the application if EDI functionality is used. Optionally 3rd party batch running infrastructure can be deployed to run batch scripts (for example, Apworks, and so on). The client workstation can be either at the corporate data center or at the retailer locations (stores and warehouses). The client workstations need to run supported browsers with optionally printers attached.

WebLogic Application Server executes server side of the application logic. Oracle Database provides data handling functionality. LDAP server provides authentication and user attributes handling. Single Sign-On infrastructure provides capability to share authentication tokens across multiple retail applications. FTP server provides delivery for EDI files. Workstation browsers execute client side application logic. Printers can be used to create physical outbound documents when EDI functionality is not used.

ReIM exchanges data with other retail application that are a part of the Oracle Retail Suite, such as RMS and with the applications that are not part of Oracle Retail Suite, such as Enterprise Business Suite (EBS).

Each component of the deployment should be secured to provide minimum required access. This includes OS security, network security, browser security, LDAP security, WebLogic security, database Security, and FTP server security. For information on how to secure each component, see the reference documentation.

Dependent Applications

Security Guides for dependent applications are found at the following Web sites:

- Oracle Database 11g Release 2:

  http://download.oracle.com/docs/cd/E11882_01/server.112/e10575.pdf

- Oracle WebLogic 10.3:

  http://download.oracle.com/docs/cd/E12840_01/wls/docs103/security.html

  Contact your vendor for other components of security guides.

ReIM Web Application Deployment

SSL

The ReIM application should be deployed using a secure HTTP endpoint. It should only be available through an "https:" URL. This requires the associated Oracle HTTP Server to be configured appropriately and to register the secure endpoint with the Single Sign-On Server. Configuration of the WebLogic hosting ReIM application should be secured as per WebLogic documentation.

Single Sign-On

In the standard supported deployment, ReIM leverages Single Sign-On infrastructure for authentication (optional). To avoid clear-text transmission of user IDs and passwords, Single Sign-On infrastructure must be configured to user TLS/SSL within all offered URLs. For specific information on configuring Single Sign-On to use TLS/SSL, see the *WebLogic Single Sign-On Administrator's Guide*.

WebLogic Wallet

The ReIM application relies on WebLogic Application Server to provide secure storage and retrieval for username and passwords required to communicate with other security conscious components or to run batches. WebLogic stores data securely in Credential Store, where Wallet is one of the possible implementations. Secure Wallet stores access credentials for database access, LDAP access, Web services access, and batch execution. ReIM application shares the wallet with WebLogic application, as WebLogic Application Server needs to store its own secure data in the wallet. ReIM will use separation partition for ReIM related credentials.

LDAP

The ReIM relies on LDAP for authentication logic. An LDAP server not only provides the storage for user data, but also serves as an authentication provider. Many LDAP implementations can define security policies that can define credential complexity, password history, unsuccessful login attempt count, and so on.

Note: Any valid ReIM user should be able to log in to LDAP. As such ReIM users should have read-only LDAP access.

Technical Overview of the Security Features

The following section describes the authentication, authorization, audit, and user management.

Security Features of the Application

The relevant security features fall into one or more of the following categories. For information on these categories, see the following sections:

- Authentication
- Authorization
- Audit
- User Management

Authentication

Only authenticated users should have access to ReIM application. By the point of application authentication the user potentially successfully authenticates with the network and/or the workstation OS.

ReIM supports the following two types of authentication:

- Dedicated
- Single Sign-On

Dedicated authentication - User credentials (username and password) are submitted by the Application Server logic to the LDAP server. If the user can successfully log in to LDAP server with provided credentials then the user has passed the first step of authentication. For the authentication to complete successfully the user should also have all the mandatory attributes defined within LDAP entity. If some mandatory attributes are missing then the authentication process will fail.

All LDAP attribute mappings are defined in the ldap.properties file. The following attributes have to be defined for a user to be able to successfully authenticate with ReIM:

- user_first_name_attribute_name

- user_last_name_attribute_name
- user_email_attribute_name
- user_password_attribute_name
- user_language_attribute_name
- user_country_attribute_name
- user_main_key

The attributes listed are not actual attribute names, but rather logical names and should be mapped to actual attribute names within ldap.properties configuration file.

Dedicated authentication credentials are either provided by the application online user in real time (entering user name and password on the ReIM provided login page) or are retrieved by the Application Server from the Secure Wallet in the case of batch process. The batch user still needs to provide credentials alias, so that appropriate username and passwords are retrieved from the Secure Wallet.

Single Sign-On authentication - In case of Single Sign-On authentication happens only once per application suite. Authentication is performed by Single Sign-On infrastructure (potentially backed by the same LDAP server as the case for dedicated authentication). In this case login page is provided by the Single Sign-On infrastructure. The second step of authentication is exactly the same as in dedicated authentication - mandatory user attributes should be present in LDAP server.

In both cases of authentication, the authentication is performed when it is determined that the session is not authenticated or the session has been invalidated (in case of dedicated authentication).

Authorization

ReIM supports the following two types of authorization:

- Enterprise
- Business

Enterprise authorization is performed by Application Server logic with the data provided by LDAP server. Only application users that belong to a dedicated ReIM user role/group are authorized for ReIM. So if two users exist in LDAP and both can successfully log in to LDAP server, but only one is a member of ReIM group/role then only such ReIM user can be authorized for ReIM. Enterprise authorization is Yes/No kind and does not provide granular access control.

Business authorization on the other hand is performed by the Application Server logic based on the data in the database. This authorization is performed at the user group level. Each authenticated user should be a member of a valid ReIM user group (defined in the database). The authorization is done against the set of privileges defined for the user group. Only the user whose user group allows to perform the desired operation would be authorized to proceed. If a user is a valid user and is a member of a ReIM group in LDAP but does not belong to any valid user group in database then such a user will not be authorized into ReIM application.

ReIM authorization is performed for each user driven operation. Business authorization is not performed for batch processes, as it is assumed that the batch user is authorized to perform all operations.

Audit

This can be performed at the application level and at the infrastructure level. Operating System (OS) can be configured to audit user access and processes invoked. Network layer can be configured to audit entire communication data set. Application Server can be configured to audit access to the application, including all URL requested. Database can be configured to audit each table separately or entire session. ReIM application has some limited auditing capabilities. Besides operational tables ReIM has audit tables that hold data that may be required for audit purposes. The audit tables are as follows:

- IM_EDIRJT_DOCDTL_ALW_AUDIT
- IM_EDI_RJT_DOC_DTL_AUDIT
- IM_EDI_RJT_DOC_HEAD_AUDIT
- IM_EDI_RJT_DOC_NM_AUDIT
- IM_ITEM_TAX_AUDIT
- IM_ITEM_VAT_AUDIT
- IM_ORDER_ITEM_TAX_AUDIT
- IM_TOLERANCE_DEPT_AUDIT
- IM_TOLERANCE_SUPP_AUDIT
- IM_TOLERANCE_SUTRT_AUDIT
- IM_TOLERANCE_SYS_AUDIT

> **Note:** For database audit the result will not identify the application user who performed the operation, but rather the database user. In case of ReIM all application users share the same database user.

ReIM audit data is stored in AUDIT tables. The application does not provide read access for the table. The tables are intended to be inspected through other means. Other kind of audit data access control is provided by the auditing component.

Additional audit can be performed by decreasing logging level of the application. In this case additional information is reported into logs. The drawback is that performance can suffer and that amount of log entries to be recorded will increase. You can selectively decrease logging (see Log4J documentation on logging configuration for more information). Also make sure that the log files generated by the application are secure and are accessible to authorized OS users only.

User Management

The ReIM does not store or maintains users. Instead ReIM relies on external LDAP user management applications to provide user management functionality on its behalf.

To create a new user to be used within ReIM application, you need to create a new user in LDAP; the user should be assigned to the 'ReIM' role within the LDAP. A new ReIM user must also be added to an ReIM group defined in the ReIM database. This is done through the User Group Maintenance screen. Only the user who is a member of a group that allows adding user to groups can add a new user to a group. The user adding a new user to group is not required to be member of the group where the new user is being added to.

In order to remove the user from ReIM, you need to first remove the user from a user group within ReIM, then remove the user from a group in LDAP, and finally delete the user from LDAP.

Encryption and Hashing

ReIM uses a cryptographically strong pseudo-random number generator algorithm to generate Cross Site Request Forgery (CSRF) token. On Windows platform SHA1PRNG is used by default and Secure Random Number (SUN) is used as strong pseudo-random number generator provider. This gives 160-bit seed. On other platforms different algorithms and providers can be used. The configuration is done in Owasp.CsrfGuard.properties.

- **org.owasp.csrfguard.PRNG**
- **org.owasp.csrfguard.PRNG.Provider**

Also JPS encrypts data stored into the Secure Wallet. For more information on the default encryption algorithm used, see *Credential Store Manager* section in WebLogic documentation.

12

Post Installation - ReIM Application Administration

This chapter provides information about application administrative tasks related to security. How to manage users and roles as well as some other common application administrative tasks such as secure credential management and logging are discussed.

The following topics are covered in this chapter:

- Roles and Permissions
- Other Common Application Administration

Roles and Permissions

ReIM uses the following two kinds of user roles:

- Enterprise
- Business

There should be only a single enterprise user role created and it is defaulted to ReIM. This role is created in LDAP by the system administrator. Any LDAP user that should be considered for ReIM processing should be made a member of this role. ReIM makes an assumption that LDAP implementation will have groups that keep track of its member and the users will not be aware about their group membership. It implies that there will be a reference maintained by the user group to the user group member by means of **role_member** mapped attribute. For example if **role_member** attribute is mapped to **uniqueMember**, then **uniqueMember** attribute will point to the group member. The user, on the other hand does not have any attributes pointing to the ReIM group. ReIM does not provide any Lightweight Directory Interchange Format (LDIF) scripts to create ReIM user role.

The script creates the following three business user groups when you run the demo data script: Admin, Demo Users, and Limited Privs Users. No user groups exists by default and the retailer is responsible for creating the groups manually by inserting data into the appropriate table (demo data script can be used as a starting point), if the script is not run. As the names suggest the Admin group has the highest level of privileges, Demo Users group has smaller set of privileges and the Limited Privs Users has very limited set of privileges. The name of the group is completely arbitrary. There can be as many groups as required. The best approach is defined as the user group based on the set of task the user group members are supposed to perform and assign only the minimum required set of privileges to that group. For example, if the user is a warehouse manager who is responsible for resolving quantity discrepancies associated with his/her warehouses then a Warehouse Manager role can be created and Quantity Discrepancy Resolution privilege assigned to it.

You need to avoid assigning all users to Administrator role. This will grant all users more privileges than it is required. The best approach is to assign new users to Limited Privs Users group and then move user to another group as needed.

It is recommended to create an Admin group and an Admin user through a script and then to use Application logic (User Group Maintenance Screen) to maintain user groups.

The application logic allows the following actions:

1. Create/delete/update an user group
2. Assign/remove users to/from a role
3. Define a role's workflow permissions
4. Assign Location Hierarchy to the role
5. Assign Merchandise Hierarchy to the role

Location and Merchandising Hierarchy data security is limited and mostly tied to discrepancy resolution functionality.

Other Common Application Administration

As a part of the operational workflow, ReIM needs to have credential information to authenticate application users or to authenticate application itself with other dependent components such as database, LDAP, or Web services. For the case of remote users connecting to ReIM servers through browsers credentials are retrieved in real time through an online form. For all other cases the credentials are determined at installation and are stored in Secure Wallet by means of Credential Store Manager Component. At runtime the credentials are retrieved from the wallet and supplied to the component for authentication. The credentials can be updated if required. As part of installation ReIM provides convenience scripts that allow credential entries to be updated. The scripts allow the system administrator to see usernames stored in ReIM wallet partition and to change the password if necessary. The script does not display original passwords. For more information, see the *Operations Guide*.

There are two set of logs that are generated by ReIM application - infrastructure logs and actual Application logs. Infrastructure logs are configured and maintained by appropriate tools for infrastructure component. An example of infrastructure log would be WebLogic application server various logs. The logging level and other logging parameters can be adjusted through WebLogic component. Such logs are owned by the OS user that own WebLogic process. WebLogic log files are located within WebLogic directory structure. Application logs are generated by ReIM application logic. ReIM log configuration is done through ReIM log configuration file. The administrator has the rights to configure the location of the ReIM log files. ReIM log files are also owned by the OS user that own WebLogic process. It is recommended to grant access to log files only to the administrators.

ReIM application does not restrict concurrent sessions from the same user. It means that more than a single user can log in to ReIM server with the same credentials. There will be more than one session from application standpoint, but there will be the same user from database standpoint of view. It is recommended not to use the same credentials for different sessions.

The session is maintained per browser instance. So if more than a single browser is used then the server will consider such scenario as multiple user logins. At the same time multiple tabs of the same browser would share the session.

Session timeout is defined at the application server level. It is 60 minutes by default, but can be changed through WebLogic configuration.

13

Extending/Customization

Customization and extending capabilities is an important part of any application. This chapter discusses how to securely implement customizations and extensions such that they do not jeopardize application security.

If customization is required it should be done in such a way that no built-in explicit security features would be circumvented. For example, all requests for a resource within ReIM application will go through CSRF filter to guarantee that required security token is present. You need to protect the page by the existing filter, if new page has to be added.

The most common form of customization is to modify Data Access Object to provide the full form of DAO) layer. ReIM has a customization hook built in. An appropriate configuration has to be done to pick custom bean instead of base bean, if Spring managed bean has to be modified. The custom classes are picked in preference or AAccess, if Access classes have to be modified (as opposite of AAccess classes).

It is recommended to perform secure code analysis after code customization to identify potential secure coding standard violations.

The customization should store those additional credentials in the Secure Wallet along with all other ReIM credentials (in ReIM partition), if additional integration and credentials are required. Credential population should be done by a script provided with ReIM.

14

Securing the Database

The database should be secured using the recommendations from the *Oracle Database 11g Release 2 Security Guide*.

The following sections provide additional application specific guidance for securing the database for use with Oracle Retail Invoice Matching application.

Application Schema Owners

As ReIM shares schema owner with RMS, you need to follow the RMS security guidelines regarding schema owner permissions.

ReIM should not use schema owner for database communication. Instead a schema synonym should be used.

Database Security Considerations

The following recommendations should be considered for the database:

- The database server should be in a private network.
- The database server should be in a locked secure facility and inaccessible to non-administrator personnel.
- The database should only be accessed through trusted network hosts.
- The database server should have minimal use of ports and any communications should be under secure protocols.
- The database should be on its own dedicated server.
- The database server should be behind a firewall.
- Any database user beyond the schema application owner should be audited.
- Only minimal rights should be granted to the owner of database processes and files such that only that owner has the right to read and write from the database related files, and no one else has the capability to read and write from such files.

The purge script is usually put into an automation script, which runs once a day. As described above, this script is usually run by a user with limited access (only execute procedure and connect access).

Restricted Access to Purge Batches

ReIM use batch infrastructure for purging data. As such user authentication is required. The purging processes should be scheduled and executing any individual data purging process outside of this schedule should be avoided.

If some additional purging is required on a regular basis that is outside of the purging functionality provided by ReIM, you need to do that through standard set of scripts that should have security built into it. You need to run the custom purging scripts under a separate schema.

Part IV

Oracle Retail Price Management (RPM)

The following chapters provide guidance for administrators, developers, and system integrators who securely administer, customize, and integrate the Oracle Retail Price Management (RPM) application.

Part IV contains the following chapters:

15

General Security Considerations

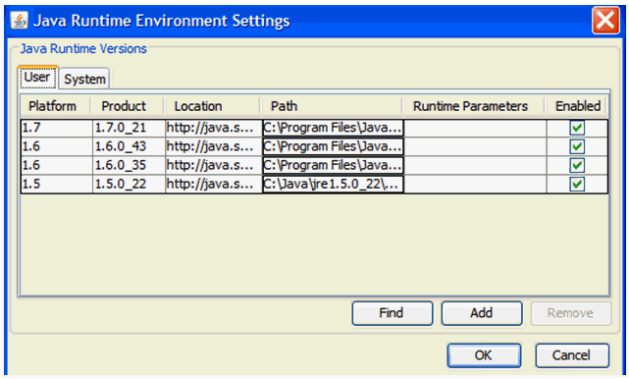
This chapter discusses how to securely install the Oracle Retail Price Management (RPM) application. To obtain a secure configuration, follow the instructions and advice provided below.

The RPM application is installed on the server; however it is used in the distributed environment. Both client and server security must be taken into consideration when hardening application deployment. You need to reference your desktop and server operation system security guides, if available for more information on reinforcing security for the execution environment.

In particular, only valid users should have access to the client workstations running clients for the application. Since the RPM user interface has no inactivity timeouts, a reasonable locking policy should be established to lock out computer screens after some time of inactivity. Security policy should be established at the desktop level to monitor unsuccessful login attempts. System administrator should guarantee that the operation system has the latest mandatory update patches.

As RPM client executes in the Java sandbox, it is important to keep Java runtime up-to-date with the latest security patches. Java runtime can be provided by either a browser plug-in or by standalone JVM such as the one shipped with JDK. You need to make sure that runtime configuration for appropriate Java version is correct. See Figure 15–1 for example on how Java runtime environment settings window looks like.

Figure 15–1 Java Runtime Environment Settings Window



Ensure the following:

- The product identifier matches actual version of the runtime pointed by path
- Disable runtimes that are not up-to-date are not required

- The configuration is correct at both system and user level (for each user on the workstation)
- The network configuration is correct and connection will go either directly or through authorized proxy
- Verify that the runtime configuration has enough temporary space allocation to download and keep RPM client (see RPM distribution for the latest client size)
- The correct certificates are installed on client's workstations. As RPM installation requires signing of distributed ears and jar files and self-generated keys are not allowed anymore, make sure that retailer certificate is used for signing and that the certificate is installed correctly
- Double-check that Java runtime is configured to allow JNLP file invocation - the best setting would be to prompt user

System administrator should restrict users from modifying security setting for Java runtime. At the same time the user should be allowed to accept JNLP security requests. Only signed trusted code should be downloaded and run. The system should check that the certificate matches hostname of the server. The retailer should be registered as a trusted publisher with the workstation. SSL should be enabled on the client.

For more information of how secure the internal network used to access RPM server(s), see *Network Security Configuration Guide.* This should include both physical and logical security of the network. Only SSL enabled communication should be used.

Only the desktops on the intranet should be allowed accessing the application server. The best approach is to limit the set of client computers on the network that can access the application server. That can be done at the network level to prevent guest users on the local network from even seeing the application server.

Only system administrators should have access to the application server(s). Business users (even power users) should not have accounts on the application server computers. If such accounts do exist, the OS account privileges should prevent business user from accessing application server files/directories associated with ReIM application.

The users running batches should not be system administrators. The best approach is to have a single dedicated user for running batches, rather than having multiple users running batches ad hoc. At the same time RPM application does not prevent any valid application user from running batches. The main difference between batch user and regular users is that batch user credentials exist in the Secure Wallet. Installer is the one responsible for placing batch user credentials into the wallet, so by default there will be only a single batch user. At the same time nothing is preventing the system admin from adding additional valid RPM users into Secure Wallet (by running provided scripts).

RPM both consumes and produces data files. File consumption is done on the client side where a user can select a file on the file system containing a Price Event Item List. Only CSV and Excel file types are allowed. It is recommended that the file transfer to the client workstation is done through secure FTP (in case the file is generated somewhere else and not on the same workstation). File generation is done on the server. It is recommended to keep the file being uploaded for audit purposes. It is up to the retailer to provide the process for that. The files can be kept in an audit directory or using any other appropriate document management system that would allow easy retrieval later on. Generated price event files are transmitted through secure ftp.

16

Understanding Security

This chapter provides an overview of all the security features available in the RPM application. It is provided as a reference for how the application securely communicates with other applications; how the application authenticates, authorizes, and audits users; and the encryption and hashing mechanisms used by the application.

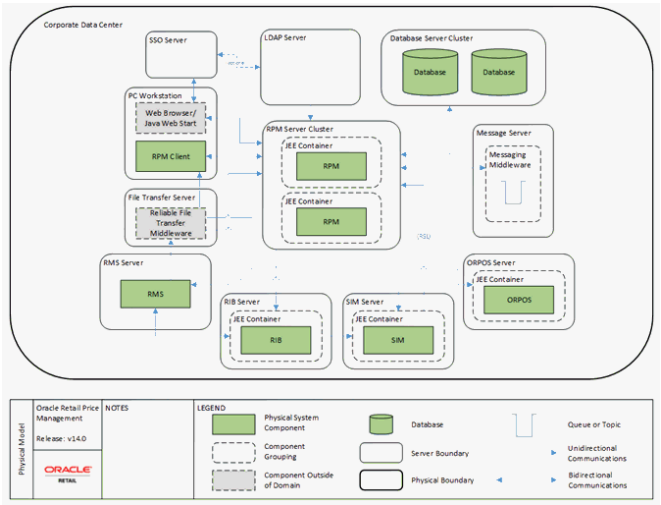
The following topics are covered in this chapter:

- Security Features Overview
- Dependent Applications
- Discussion of Dependencies on Underlying Platform
- Technical Overview of the Security Features
- Security Features of the Application
- Encryption and Hashing

Security Features Overview

Figure 16–1 shows the physical deployment of RPM.

Figure 16–1 RPM Physical Deployment



The Application server is deployed on the corporate intranet on WebLogic cluster. The Application server is connected to a cluster of Oracle databases and a corporate LDAP

server. Optionally Single Sign-On infrastructure can be deployed. In addition secure FTP server is responsible for transferring files in/out of the application on the server and on the client (if required). Optionally 3rd party batch running infrastructure can be deployed to run batch scripts (for example, Apworks, and so on). The client workstation should be at the corporate data center. The client workstations need to run supported Java runtime.

WebLogic Application server executes server side of the application logic. Oracle Database provides data handling functionality. An LDAP server provides user attributes handling. Single Sign-On infrastructure provides capability to share authentication tokens across multiple retail applications. An FTP server provides delivery for incoming/outgoing files. Workstation Java runtime executes client side application logic.

RPM is exchanging data with other retail application that are a part of the Oracle Retail Suite, such as RMS, SIM, and POS.

Each component of the deployment should be secured to provide minimum required access. This includes OS security, network security, browser security, LDAP security, WebLogic security, database Security and FTP server security. For information on how to secure each component, see the reference documentation.

Dependent Applications

Security Guides for dependent applications are found at the following Web sites:

- Oracle Database 11g Release 2:

  http://download.oracle.com/docs/cd/E11882_01/server.112/e10575.pdf

- Oracle WebLogic 10.3:

  http://download.oracle.com/docs/cd/E12840_01/wls/docs103/security.html

Contact your vendor for other components of security guides.

Discussion of Dependencies on Underlying Platform

RPM relies on Java runtime on the client workstations to provide client side code execution.

RPM replies on WebLogic to provide authorization for accessing application services. Only authenticated authorized users should be allowed to access Enterprise JavaBeans (EJBs) and data sources installed as part of RPM installation on WebLogic servers. For more information, see the Authorization section.

Technical Overview of the Security Features

The following section describes the authentication, authorization, audit, and user management.

Security Features of the Application

The relevant security features fall into one or more of the following categories. For information on these categories, see the following sections:

- Authentication
- Authorization
- Audit

- User Management

Authentication

Only authentication users should have access to RPM application. By the point of application authentication the user potentially successfully authenticates with the network and/or the workstation OS.

RPM supports the following two types of authentication:

- Dedicated
- Single Sign-On

Dedicated authentication - User credentials (username and password) are verified by WebLogic Authenticator. For information on registering an Authenticator with WebLogic, see the *Oracle Retail Price Management Installation Guide*. LDAP authenticator should be used where actual authentication data is provided by LDAP.

> **Note:** Even though LDAP provides the data, the actual verification of provided credentials is done by WebLogic.

If the WebLogic has been successfully verified the user credentials then the user has passed the first step of authentication. For the authentication to complete successfully the user should also have all the mandatory attributes defined within LDAP entity. If some mandatory attributes are missing then the authentication process will fail.

All LDAP attribute mappings are defined in the security.properties file. The following attributes (besides user identifier attribute) have to be defined for a user to be able to successfully authenticate with RPM:

- ldap.firstname.attrname
- ldap.lastname.attrname

The attributes listed are not actual attribute name, but rather logical name and should be mapped to actual attribute name within security.properties configuration file.

Dedicated authentication credentials are either provided by the application online user in real time (entering the user name and password on the RPM provided login page) or are retrieved by the application server from the Secure Wallet in the case of batch process. The batch user still needs to provide credentials alias, so that appropriate username and passwords are retrieved from the Secure Wallet.

Single Sign-On authentication - In case of Single Sign-On authentication happens only once per application suite. Authentication is performed by Single Sign-On infrastructure (potentially backed by the same LDAP server as the case for dedicated authentication). In this case login page is provided by the Single Sign-On infrastructure. The second step of authentication is exactly the same as in dedicated authentication - mandatory user attributes should be present in LDAP server.

In both cases of authentication, the authentication is performed when it is determined that the session is not authenticated or the session has been invalidated (in case of dedicated authentication).

Authorization

RPM supports the following two types of authorization:

- Enterprise

- Business

Enterprise authorization - This is performed by application server logic. WebLogic will make sure (if configured correctly - see the *Oracle Retail Price Management Installation Guide* for more information on required WebLogic configuration) that already authenticated user is the member of RPM user role/group (Role is created LDAP). Only the users who are members of RPM groups are allowed to get access to RPM functionality. All EJBs and data sources are accessible to RPM Secure User.

When the user submits a request to the server, the request (after authentication) is handled by a Controller EJB. This controller EJB will determine appropriate EJB to perform the action. When controller EJB will issue another request to Worker EJB, this request will be performed under RPM Secure User principal. Same discussion applies to accessing Data sources.

Business authorization - This is performed by Oracle Retail Security Management (RSM) logic. RSM is a part of RPM distribution. RMS will define what action each user role can perform and in what mode those action can be performed (allowed modes are Edit, View, None, Emergency). In addition to action security RSM provides ability to define data security. Data security allows system to associate appropriate level of location and merchandising hierarchy with a user role. Note that RSM operates on the user role level and not on the user level.

Audit

RPM does not provide extensive audit capabilities. Only price even tables carry the user information. The user information is relevant only as the creation identifier and does not guarantee that the final price event was completed by the same user who created the price event. Hence if auditing is required, it should be done at the database level. For information on how to turn on the database auditing, see the Oracle database documentation. The database auditing will be not be able to differentiate application users modifying data, as it is done at the database user level and all RPM application users share the same database user.

User Management

RPM does not store or maintain users. Instead RPM relies on LDAP to provide user management functionality on its behalf.

To create a new user to be used within RPM application, a new user should be created in LDAP; the user should be assigned to LDAP RPM role. Through backend the newly created user should be associated with a user role. If a new user role is required, it should be created through RSM screen first. It is recommended to create user roles with the minimum set of user privileges required for the user.

In order to remove the user from RPM, you need to first remove the user from the user group within RSM, then remove the user from a group in LDAP, and finally delete the user from LDAP.

Encryption and Hashing

This section covers securing the applications using encryption and hashing.

RPM uses encryptions as part of a security key generated on the server to deliver a client to the user workstation. To access RMS the user issues a requests to the server for Java Network Launch Protocol (JNLP) file. The server will generate temporary token based on the encryption method configured in the security.properties file.

user.signature.cipher.algorithm

By default HmacSHA1 is used, but the algorithm can be changed if required.

17

Post Installation - Application Administration

This chapter provides information about application administrative tasks related to security. How to manage users and roles as well as some other common application administrative tasks such as secure credential management and logging are discussed.

This chapter discusses post installing the application administration.

The following topics are covered in this chapter:

Roles and Permission Grants

RPM uses the following two kinds of user roles:

- Enterprise
- Business

There should be only a single enterprise user role created and it is defaulted to rpm_secure_users. This role is created in LDAP by the system administrator. Any LDAP user that should be considered for RPM processing should be made a member of this role. RPM makes an assumption that LDAP implementation will have groups that keep track of its member and the users will not be aware about their group membership. It implies that there will be a reference maintained by the user group to the user group member by means of a static member DN attribute. For example if this attribute is **uniqueMember**, then **uniqueMember** attribute will point to the group member. The user, on the other hand does not have any attributes pointing to the rpm_secure_users group. RPM does not provide any LDIF scripts to create the rpm_secure_users role.

If the data seeding script has been run, then the script would create a single business user role - Administrator Role. If the script has not been run, then no user roles exist by default and the retailer is responsible for creating the groups manually by inserting data into appropriate table (demo data script can be used as a starting point). The role Administrator Role, as the name suggests, has the highest level of privileges. The name of the group is completely arbitrary. Retailers can create as many additional roles as they require. The best approach when creating a user role is to define it based on the set of task the role members are supposed to perform and assign only the minimum required set of privileges to that group. For example, an employee can be part of a group which only allows the submission of price events and does not allow direct approval. This allows users with approval privileges the chance to review submitted price events and decide whether or not to approve them.

Despite it being the only role created by the data seed script, try to avoid assigning all users to Administrator Role. This will grant all users more privileges than is likely required. Create user roles that best fit the user's requirements and assign them only the privileges they require.

It is recommended to create the Administrator Role and an Admin user through a script and then to use Application logic (RSM Role Administration Screen) to maintain user roles.

The application logic allows the following actions:

1. Create/delete/update an user group
2. Assign/remove users to/from a role
3. Define a role's workflow permissions
4. Assign Location Hierarchy to the role
5. Assign Merchandise Hierarchy to the role

You cannot delete a user role from the UI, however it is possible to remove all the privileges from the role effectively eliminating it.

Other Common Application Administration

As a part of the operational workflow, RPM needs to have credential information to authenticate application users or to authenticate application itself with other dependent components such as database, LDAP or Oracle Retail Integration Bus (RIB). For the case of remote users connecting to RPM servers through RMI credentials are retrieved in real time through client provided form. For all other cases the credentials are determined at installation and are stored in Secure Wallet by means of Credential Store Manager Component. At runtime the credentials are retrieved from the wallet and supplied to the component for authentication. The credentials can be updated if required. As part of installation RPM provides convenience scripts that allow credential entries to be updated. The scripts allow the system administrator to see usernames stored in RPM wallet partition and to change the password if necessary. The script does not display original passwords. For more information, see the *Operations Guide*.

There are two set of logs that are generated by RPM application - infrastructure logs and actual Application logs. Infrastructure logs are configured and maintained by appropriate tools for infrastructure component. An example of infrastructure log would be WebLogic application server various logs. The logging level and other logging parameters can be adjusted through WebLogic component. Such logs are owned by the OS user that own WebLogic process. WebLogic log files are located within WebLogic directory structure. Application logs are generated by RPM application logic. RPM log configuration is done through RPM log configuration file (on the client and on the server). The administrator has the rights to configure the location of the RPM log files. RPM log files are also owned by the OS user that own WebLogic process (on the server) and OS user running client Java runtime (on the client workstation). It is recommended to grant access to log files only to the administrators.

RPM application does not restrict concurrent sessions from the same user. It means that more than a single user can log in to RPM server with the same credentials. There will be more than one session from application standpoint, but there will be the same user from database standpoint of view. It is recommended not to use the same credentials for different sessions.

Session timeout is defined at the application server level. It is 60 minutes by default, but can be changed through WebLogic configuration.

18

Extending/Customization

Customization and extending capabilities is an important part of any application. This chapter discusses how to securely implement customizations and extensions such that they do not jeopardize application security.

If customization is required it should be done in such a way that no built-in explicit security features would be circumvented. Customization should be done using provided customization toolkit.

It is recommended to perform secure code analysis after code customization to identify potential secure coding standard violations.

If additional integration is required and credentials are required then the customization should store those additional credentials in the Secure Wallet along with all other RPM credentials (in RPM partition). Credential population should be done by a script provided with RPM. If additional EJB are required, they should be protected by the same authorization as pre-existing EJBs. For more information, see the *Oracle Retail Price Management Installation* Guide.

19

Securing the Database

The database should be secured using the recommendations from the *Oracle Database 11g Release 2 Security Guide*.

The following sections provide additional application specific guidance for securing the database for use with Oracle Retail Invoice Matching application.

Application Schema Owners

As RPM shares schema owner with RMS, follow RMS security guidelines regarding schema owner permissions.

RPM should not use schema owner for database communication. Instead a schema synonym should be used.

Database Security Considerations

The following recommendations should be considered for the database:

- The database server should be in a private network.
- The database server should be in a locked secure facility and inaccessible to non-administrator personnel.
- The database should only be accessed through trusted network hosts.
- The database server should have minimal use of ports and any communications should be under secure protocols.
- The database should be on its own dedicated server.
- The database server should be behind a firewall.
- Any database user beyond the schema application owner should be audited.
- Only minimal rights should be granted to the owner of database processes and files such that only that owner has the right to read and write from the database related files, and no one else has the capability to read and write from such files.

The purge script is usually put into an automation script, which runs once a day. As described above, this script is usually run by a user with limited access (only execute procedure and connect access).

Restricted Access to Purge Batches

RPM will use batch infrastructure for purging data. As such the user authentication is required. The purging processes should be scheduled and executing any individual data purging process outside of this schedule should be avoided.

If some additional purging is required on a regular basis that is outside of the purging functionality provided by RPM, do that through standard set of scripts that should have security built into it. Run the custom purging scripts under a separate schema.

Part V

Oracle Retail Allocation

This Security Guide is for users and administrators of Oracle Retail Allocation. This includes merchandisers, buyers, business analysts, and administrative personnel.

Part V contains the following chapter:

- Setting up Functional Security

20

Setting up Functional Security

The chapter provides guidance for administrators to understand, configure, and customize functional security for the Oracle Retail Allocation application.

The following topics are covered in this chapter:

Understanding the Security Model

The Oracle Retail Fusion security model is built upon the Oracle Fusion Middleware platform, which incorporates the Java security model. The Java model is a role-based, declarative model that employs container-managed security where resources are protected by roles that are assigned to users. However, extensive knowledge of the Java-based architecture is unnecessary when using the Oracle Fusion Middleware Security model.

Key Security Elements

The Oracle Fusion Middleware security model depends upon the following key elements in order to provide uniform security and identity management across the enterprise:

- **Application policy**

  Application permissions are granted to members of its application roles. In the default security configuration, each application role conveys a predefined set of permissions. Permission grants are defined and managed in an application policy. After an application role is associated with an application policy, that role becomes a Grantee of the policy. An application policy is specific to a particular application.

- **Application role**

  After permission grants are defined in an application policy, an application role can be mapped to that policy, and the application role then becomes the mechanism to convey the permissions. In this manner an application role becomes

the container that grants permissions to its members. The permissions become associated with the application role through the relationship between policy and role. After groups are mapped to an application role, the corresponding permissions are granted to all members equally. Membership is defined in the application role definition. Application roles are assigned in accordance with specific conditions and are granted dynamically based on the conditions present at the time authentication occurs. More than one group can be members of the same application role.

- **Authentication provider**

  An authentication provider is used to access user and group information and is responsible for authenticating users. An Oracle WebLogic Server authentication provider enables you to manage users and groups in one place.

  An identity store contains user name, password, and group membership information. An authentication provider accesses the data in the identity store and authenticates against it. For example, when a user name and password combination is entered at login, the authentication provider searches the identity store to verify the credentials provided. The Oracle Retail Fusion application's default authentication provider authenticates against Oracle Internet Directory (OID).

- **Users and groups**

  A user is an entity that can be authenticated. A user can be a person, such as an application user, or a software entity, such as a client application. Every user is given a unique identifier.

  Groups are organized collections of users that have something in common. Users should be organized into groups with similar access needs in order to facilitate efficient security management.

- **Security realm**

  During installation an Oracle WebLogic Server domain is created and Oracle Retail Fusion application is installed into that domain. The Oracle Retail Fusion application security is managed within the security realm for this Oracle WebLogic Server domain. A security realm acts as a scoping mechanism. Each security realm consists of a set of configured security providers, users, groups, security roles, and security policies. Only one security realm can be active for the domain. The Oracle Retail Fusion application's authentication is performed by the authentication provider configured for the default security realm for the WebLogic Server domain in which it is installed. Oracle WebLogic Server Administration Console is the administration tool used for managing an Oracle WebLogic Server domain.

Permission Grants and Inheritance

Each Oracle Retail Fusion application provides application-specific permissions for accessing different features. Application permissions are typically granted by becoming a member in an application role. Permissions can be granted in two ways - through membership in an application role (direct) and through group and role hierarchies (inheritance). Application role membership can be inherited by nature of the application role hierarchy. In the default security configuration, each application role is pre configured to grant a predefined set of permissions. Groups are mapped to an application role. The mapping of a group to a role conveys the role's permissions to all members of the group. In short, permissions are granted in Oracle Retail Fusion applications by establishing the following relationships:

- A group defines a set of users having similar system access requirements. Users are added as members to one or more groups according to the level of access required.
- Application roles are defined to represent the role a user typically performs when using the Oracle Retail Fusion application.
- The groups of users are mapped to one or more application roles that match the type of access required by the population.
- An application role is mapped to the application policy that grants the set of permissions required by the role type (an administrator, an author, a consumer).
- Group membership can be inherited by nature of the group hierarchy. Application roles mapped to inherited groups are also inherited, and those permissions are likewise conveyed to the members.

How a user's permissions are determined by the system are as follows:

1. A user enters credentials into a Web browser at login. The user credentials are authenticated by the authentication provider against data contained in the identity store.
2. After successful authentication, a Java subject and principal combination is issued, which is populated with the user name and a user's groups.
3. A list of the user's groups is generated and checked against the application roles. A list is created of the application roles that are mapped to each of the user's groups.
4. A user's permission grants are determined from knowing which application roles the user is a member of.

A user can also be granted permissions if they inherit other application roles. Members of application roles can include other groups and application roles. The result is a hierarchical role structure where permissions can be inherited in addition to being explicitly granted. This hierarchy provides that a group is granted the permissions of the application role for which it is a member, and the permissions granted by all roles descended from that role.

For example, the default security configuration includes several predefined groups and application roles. The default BIAdministrator application role includes the BIAdministrators group, the BIAuthor application role includes the BIAuthors group, and the BIConsumer application role includes the BIConsumers group. The default BIAdministrator application role is a member the BIAuthor application role, and the BIAuthor application role is a member of the BIConsumer application role. The members of these application roles inherit permissions as follows. Members of the BIAdministrators group are granted all the permissions of the BIAdministrator role, the BIAuthor role, and the BIConsumer role. By nature of this role hierarchy, the user who is a member of a particular group is granted permissions both explicitly and through inheritance.

Note: By themselves, groups and group hierarchies do not enable any privilege to access resources controlled by an application. Privileges are conveyed by the permission grants defined in an application policy. A group or application role becomes a Grantee of the application policy. The application policy Grantee conveys the permissions and this is done by becoming a member of the Grantee (application role).

Figure 20–1 shows these relationships between the default groups and application roles.

Figure 20–1 Relationships between the Default Groups and Application Roles

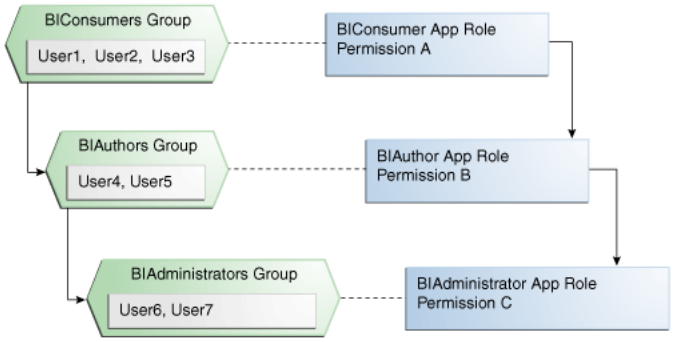


Table 20–1 summarizes how permissions are granted explicitly or are inherited in the previous example and in Figure 20–1.

Table 20–1 Permissions Granted by the Role Hierarchy Example

User Name	Group Membership: Explicit/Inherited	Application Role Membership: Explicit/Inherited	Permission Grants: Explicit/Inherited
User1, User2, User3	BIConsumers: Explicit	BIConsumer: Explicit	Permission A: Explicit
User4, User5	BIAuthors: Explicit	BIAuthor: Explicit	Permission B: Explicit
	BIConsumers: Inherited	BIConsumer: Inherited	Permission A: Inherited
User6, User7	BIAdministrators: Explicit	BIAdministrator: Explicit	Permission C: Explicit
	BIAuthors: Inherited	BIAuthor: Inherited	Permission B: Inherited
	BIConsumers: Inherited	BIConsumer: Inherited	Permission A: Inherited

Default Security Configuration

Access control of system resources is achieved by requiring users to authenticate at login and by restricting users to only those resources for which they are authorized. A default security configuration is available for immediate use after the Oracle Retail Fusion application is installed and is configured to use the Oracle Fusion Middleware security model. The default configuration includes three predefined security roles for application specific permission grants. Users can be added to predefined groups that are mapped to preconfigured application roles. Allocation is preconfigured to grant specific application permissions.

Table 20–2 Privileges

Name	Description
Search Allocations Priv	A privilege for searching for allocations.
Maintain Allocation Priv	A privilege for creating and editing and an allocation.
Delete Allocation Priv	A privilege for deleting an allocation.
View Allocation Priv	A privilege for viewing an allocation.

Table 20–2 (Cont.) Privileges

Name	Description
Submit Allocation Priv	A privilege for submitting an allocation for approval.
Review Allocation Priv	A privilege for approving or reserving an allocation.
Batch Allocation Priv	A privilege for running batch jobs.
Search Allocation Location Groups Priv	A privilege for searching for allocations.
Maintain Allocation Location Group Priv	A privilege for creating and editing and an allocation location group.
Delete Allocation Location Group Priv	A privilege for deleting an allocation location group.
View Allocation Location Group Priv	A privilege for viewing an allocation location group.
Search Allocation Policy Templates Priv	A privilege for searching for allocations.
Maintain Allocation Policy Template Priv	A privilege for creating and editing a Policy Template.
Delete Allocation Policy Template Priv	A privilege for deleting a Policy Template.
View Allocation Policy Template Priv	A privilege for viewing a Policy Template.
Search Size Profile Priv	A privilege for searching Size Profiles.
Maintain Size Profile Priv	A privilege for creating and editing and a Size Profile.
Delete Size Profile Priv	A privilege for deleting a Size Profile.
View Size Profile Priv	A privilege for viewing a Size Profile.
Maintain System Options System Properties Priv	A privilege for editing the System Properties for System Options.
Maintain System Options User Group Properties Priv	A privilege for editing the user group properties for System Options.
View System Options Priv	A privilege for viewing System Options.

Table 20–3 Duties

Duty	Description	List of Privileges
Allocation Management Duty	A duty for managing allocations. This duty is an extension of the Allocation Inquiry Duty.	All privileges found in the Allocation Inquiry Duty. Maintain Allocation Priv, Delete Allocation Priv,
Allocation Inquiry Duty	A duty for viewing allocations.	View Allocation Priv, Search Allocations Priv,
Allocation Submit Duty	A duty for submitting allocation for approval.	Submit Allocation Priv,
Allocation Review Duty	A duty for approving or rejecting an allocation.	Review Allocation Priv,
Allocation Batch Duty	A duty for running batch process.	Batch Allocation Priv,

Table 20–3 (Cont.) Duties

Duty	Description	List of Privileges
Allocation Location Groups Management Duty	A duty for managing allocation location groups. This duty is an extension of the Allocation Location Groups Inquiry Duty and Allocation Location Group Search Duty.	All privileges found in the Allocation Location Groups Inquiry Duty and the Allocation Location Groups Search Duty. Maintain Allocation Location Groups Priv, Delete Allocation Location Groups Priv,
Allocation Location Groups Inquiry Duty	A duty for viewing allocation location groups.	View Allocation Location Groups Priv, Search Allocation Location Groups Priv,
Allocation Policy Template Management Duty	A duty for managing allocation policy template. This duty is an extension of the Allocation Policy Template Inquiry Duty and Allocation Policy Template Search Duty.	All privileges found in the Allocation Policy Template Inquiry Duty and the Allocation Policy Template Search Priv, Maintain Allocation Policy Template Priv, Delete Allocation Policy Template Priv,
Delete Allocation Location Group Priv	A duty for viewing allocation Policy Template.	View Allocation Policy Template Priv, Search Allocation Policy Template Priv,
Size Profile Management Duty	A duty for managing size profile. This duty is an extension of the Size Profile Inquiry Duty.	All privileges found in the Size Profile Inquiry Duty. Maintain Size Profile Priv, Delete Size Profile Priv,
Size Profile Inquiry Duty	A duty for viewing allocation Policy Template.	View Size Profile Priv, Search Size Profiles Priv,
System Options System Properties Management Duty	A duty for managing the system properties in system options. This duty is an extension of the System Options Inquiry Duty.	All privileges found in the System Options Inquiry Duty. Maintain System Options System Properties Priv,
System Options User Group Properties Management Duty	A duty for managing user group properties system options. This duty is an extension of the System Options Inquiry Duty.	All privileges found in the System Options Inquiry Duty. Maintain System Options User Group Properties Priv,
System Options Inquiry Duty	A duty for inquiring on profile. This duty is an extension of the Size Profile Inquiry Duty.	All privileges found in the System Options Inquiry Duty. Maintain System Options Priv,

Table 20–4 Function Security Mapping

Role	Duty	Privileges
Administrator	Allocation Management Duty, Allocation Submit Duty, Allocation Review Duty, Allocation Location Groups Management Duty, Allocation Policy Template Management Duty, Size Profile Management Duty, System Options System Properties Management Duty, System Options User Group Properties Management Duty, Allocation Batch Duty.	Search Allocations Priv, Maintain Allocation Priv, Delete Allocation Priv, Submit Allocation Priv, Review Allocation Priv, View Allocation Priv, Search Allocation Priv, View Allocation Location Groups Priv, Search Allocation Location Groups Priv, View Allocation Policy Template Priv, Search Allocation Policy Templates Priv, View Size Profile Priv, Search Size Profile Priv, View System Options Priv, Maintain System Options User Group Properties Priv, Batch Allocation Priv,
Allocation Manager	Allocation Management Duty, Allocation Submit Duty, Allocation Review Duty, Allocation Location Groups Management Duty, Allocation Policy Template Management Duty, Size Profile Management Duty, System Options User Group Properties Management Duty.	Search Allocations Priv, Maintain Allocation Priv, Delete Allocation Priv, Submit Allocation Priv, Review Allocation Priv, View Allocation Priv, Search Allocation Priv, View Allocation Location Groups Priv, Search Allocation Location Groups Priv, View Allocation Policy Template Priv, Search Allocation Policy Templates Priv, View Size Profile Priv, Search Size Profile Priv, View System Options Priv, Maintain System Options User Group Properties Priv,
Allocator	Allocation Management Duty, Allocation Submit Duty, Allocation Review Duty, Allocation Location Groups Inquiry Duty, Allocation Policy Template Inquiry Duty, Size Profile Management Duty, System Options Inquiry Duty.	Search Allocations Priv, Maintain Allocation Priv, Delete Allocation Priv, Submit Allocation Priv, Review Allocation Priv View Allocation Priv, Search Allocation Priv, View Allocation Location Groups Priv, Search Allocation Location Groups Priv, View Allocation Policy Template Priv, Search Allocation Policy Templates Priv, View Size Profile Priv, Search Size Profile Priv, View System Options Priv,
Buyer	Allocation Inquiry Duty, Allocation Policy Template Inquiry Duty, Allocation Location Groups Inquiry Duty, Allocation Size Profile Inquiry Duty.	View Allocation Priv, Search Allocation Priv, View Allocation Location Groups Priv, View Allocation Policy Template Priv, Search Size Profile Priv, View Size Profile Priv,

For more information about the Oracle Fusion Middleware security model and the authenticated role, see *Oracle Fusion Middleware Application Security Guide*.

Managing Authorization

After a user is authenticated, further access to application resources is controlled by the granting of permissions, also known as authorization. The policy store contains the system and application-specific policies and roles required for an application. A policy store can be file-based, LDAP-based or Oracle database and holds the mapping definitions between the default Oracle Retail Fusion Application's application roles, permissions and groups. Oracle Retail Fusion Application's permissions are granted by mapping groups from the identity store to application roles and permission grants located in the policy store. These mapping definitions between groups (identity store) and the application roles (policy store) are also kept in the policy store.

Note: The best practice is to map groups instead of individual users to application roles. Controlling membership in a group reduces the complexity of tracking access rights for multiple individual users. Group membership is controlled in the identity store.

The system-jazn-data.xml file is installed and configured as the default policy store. You can continue to use the default store and modify it as needed for your environment, or you can migrate its data to an Oracle database.

The policy store and credential store must be of the same type in your environment. That is, both must be either file-based or LDAP-based or Oracle database.

Permissions must be defined in a manner that Oracle Retail Fusion Application understands. All valid Oracle Retail Fusion Application permissions are premapped to application policies, which are in turn premapped to the default application roles. You cannot create new permissions in the policy store. However, you can customize the default application policy permission grants and application role mappings as well as create your own.

Accessing Oracle Enterprise Manager Fusion Middleware Control

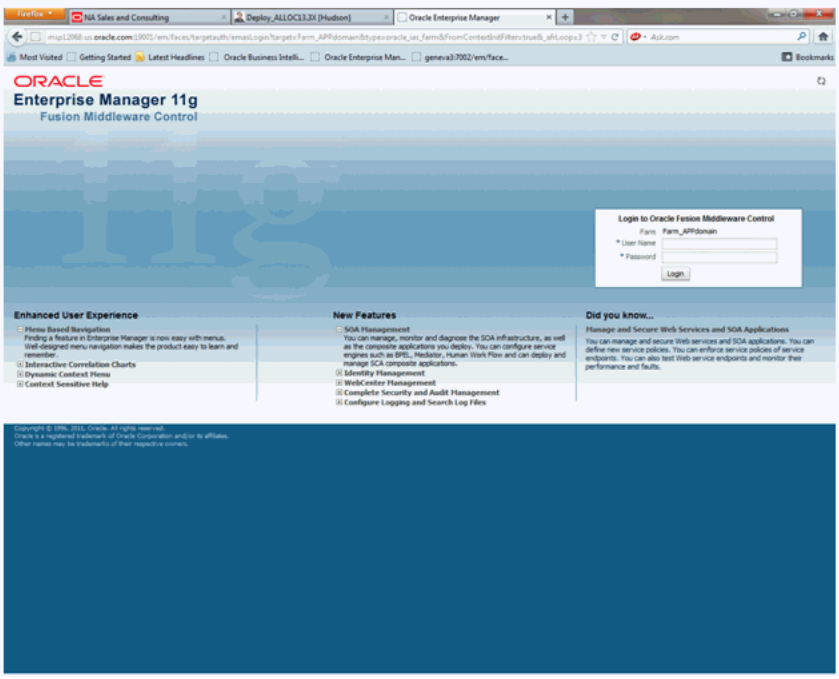
Launch Fusion Middleware Control by entering its URL into a Web browser. The URL includes the name of the host and the administration port number assigned during the installation. This URL takes the following form: http://hostname:port_number/em. The default port is 7001. For more information about using Fusion Middleware Control, see *Oracle Fusion Middleware Administrator's Guide*.

To display the Security menu in Fusion Middleware Control

1. Log on to Oracle Enterprise Manager Fusion Middleware Control by entering the URL in a Web browser. For example, http://hostname:7001/em

   The Fusion Middleware Control login page is displayed.

Figure 20–2 Fusion Middlware Control Login Page



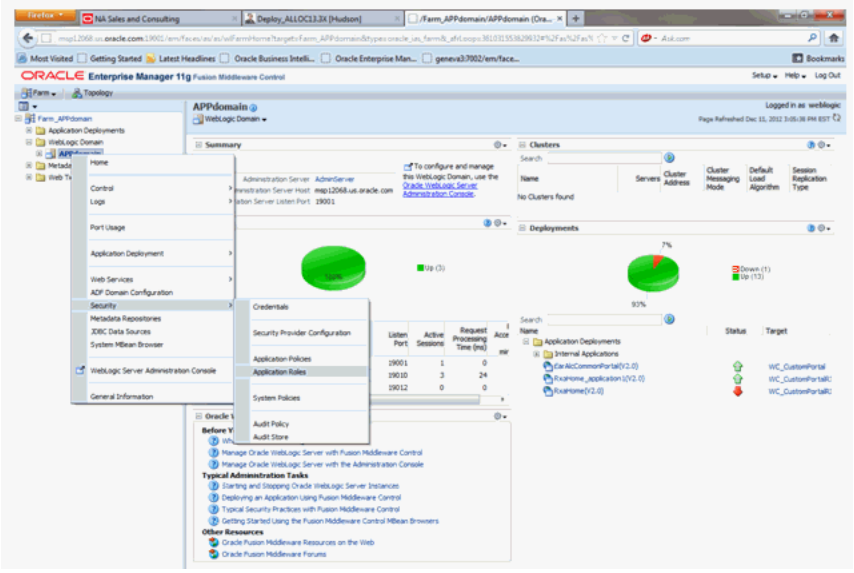
2. Enter the Oracle Retail Fusion Application's administrative user name and password and click **Login**.

   The password is the one you supplied during the installation of Oracle Retail Fusion Application. If these values have been changed, then use the current administrative user name and password combination.

3. From the target navigation pane, click **WebLogic Domain** to display **APPdomain**. Display the **Security** menu by selecting one of the following methods:

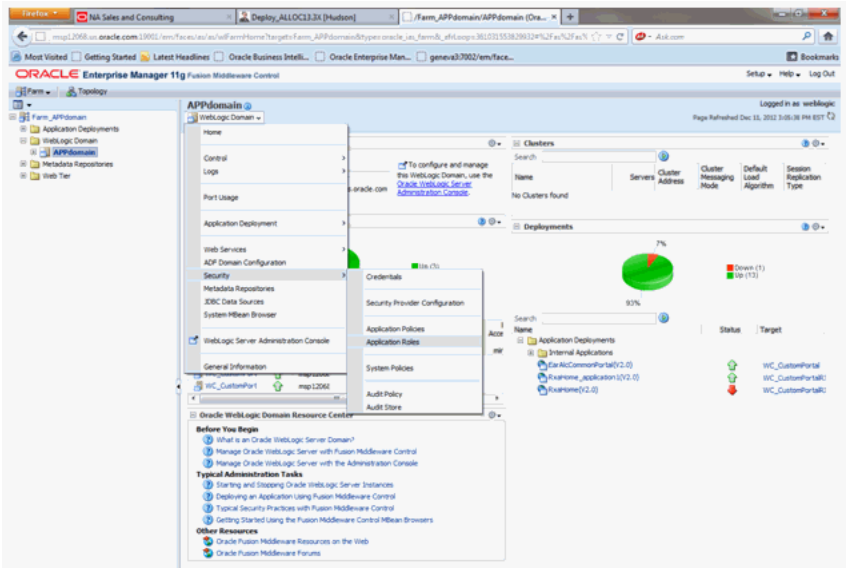
   - Right click **APPdomain** to display the Security menu.
   - Select **Security** to display a submenu.

Figure 20–3 Enterprise Manager AppDomain Security Submenu



- From the content pane, go to **WebLogic Domain** menu and select **Security**
- Select **Security** to display a submenu.

Figure 20–4 Enterprise Manager WebLogic Domain Security Submenu



Managing the Policy Store Using Fusion Middleware Control

Use Fusion Middleware Control to manage the Oracle Retail Fusion Application's application policies and application roles maintained in the policy store whether it is file-based or LDAP-based or Oracle database.

Caution: Oracle recommends you to make a copy of the original system-jazn-data.xml policy file and place it in a safe location. Use the copy of the original file to restore the default policy store configuration, if needed. Changes to the default security configuration may lead to an unwanted state. The default installation location is MW_HOME/user_projects/domain/your_domain/config/fmwconfig.

The following are common policy store management tasks:

- Modifying the membership of an application role. For more information, see Modifying Application Roles Using Fusion Middleware Control section.
- Creating a new application role from the beginning. For more information, see To create a new application role section.
- Creating a new application role based on an existing application role. For more information, see To create an application role based on an existing one section.

Modifying Application Roles Using Fusion Middleware Control

Members can be added or deleted from an application role using Fusion Middleware Control. You must perform these tasks while in the WebLogic Domain that Oracle Retail Fusion Application is installed in.

Caution: You need to be careful when changing the permission grants and membership for the default application roles. Changes could result in an unusable system.

Valid members of an application role are groups, or other application roles. The process of becoming a member of an application role is called mapping. That is, being mapped to an application role is to become a member of an application role. The best practice is to map groups instead of individual users to application roles for easier maintenance.

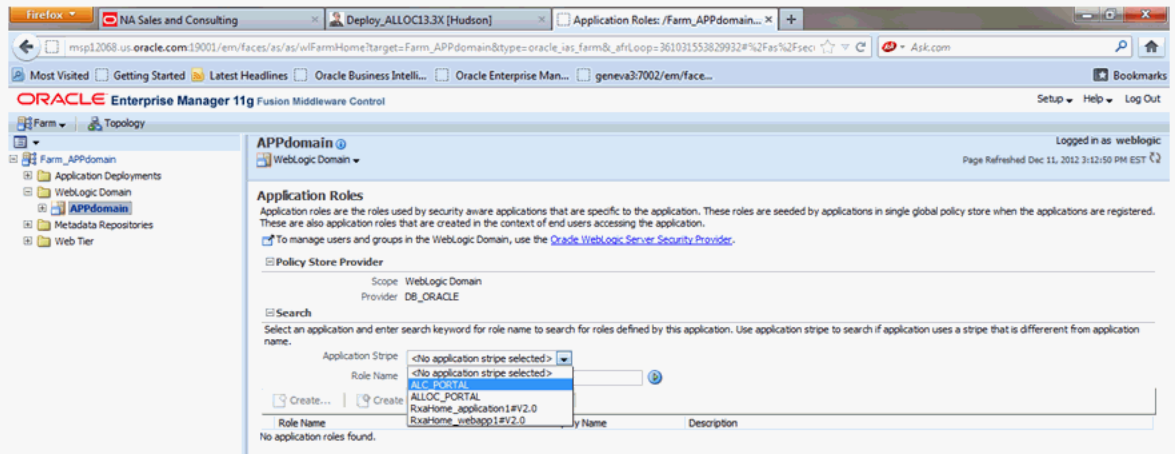
To add or remove members from an application role

1. Log in to Fusion Middleware Control, navigate to **Security**, then select **Application Roles** to display the **Application Roles** page.

   For information on navigating to the Security menu, see Accessing Oracle Enterprise Manager Fusion Middleware Control section.

2. Choose **Select Application Stripe to Search**, then select the policy stripe name (example, ALC_PORTAL) from the list.
3. Click the search icon next to **Role Name**.

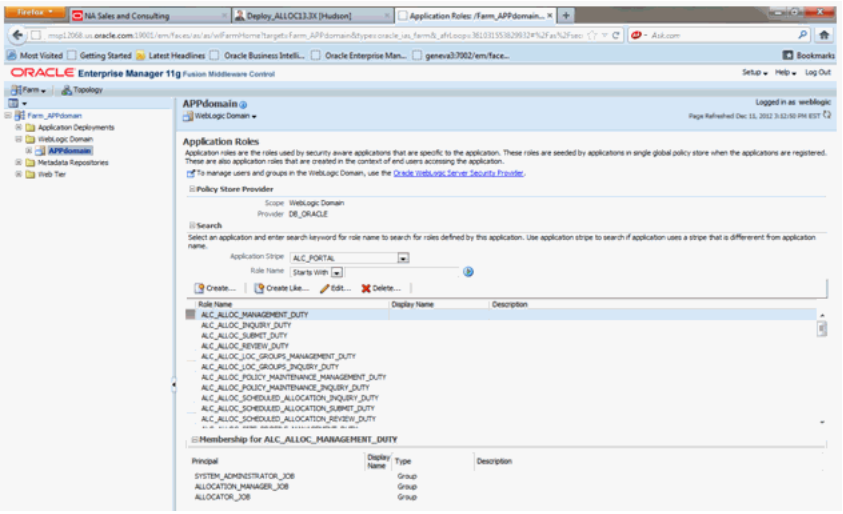
Figure 20–5 Retail Fusion Application's Application Roles Window



The Oracle Retail Fusion Application's application roles are displayed.

Figure 20–6 displays the default application roles.

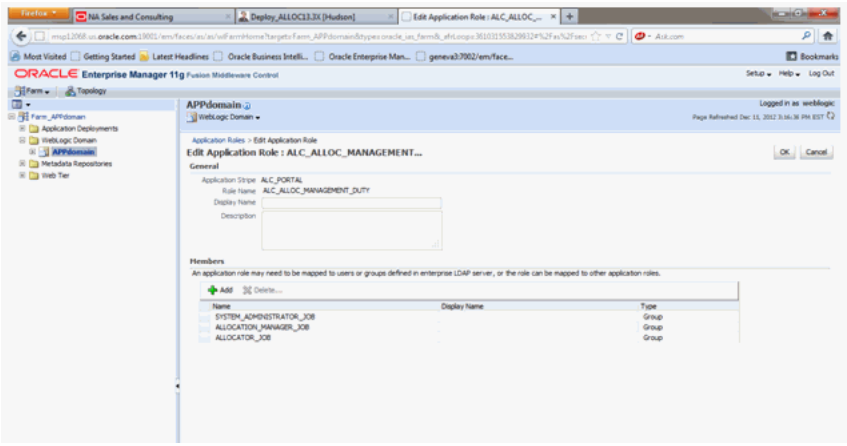
Figure 20–6 Default Application Roles Window



4. Select the cell next to the application role name and click **Edit** to display the **Edit Application Role** page.

   Figure 20–7 shows ALC_ALLOC_MANAGEMENT_DUTY role is selected.

Figure 20–7 Edit Application Role Window



Note: You can add or delete members from the **Edit Application Role** page. The valid members are application roles, and groups

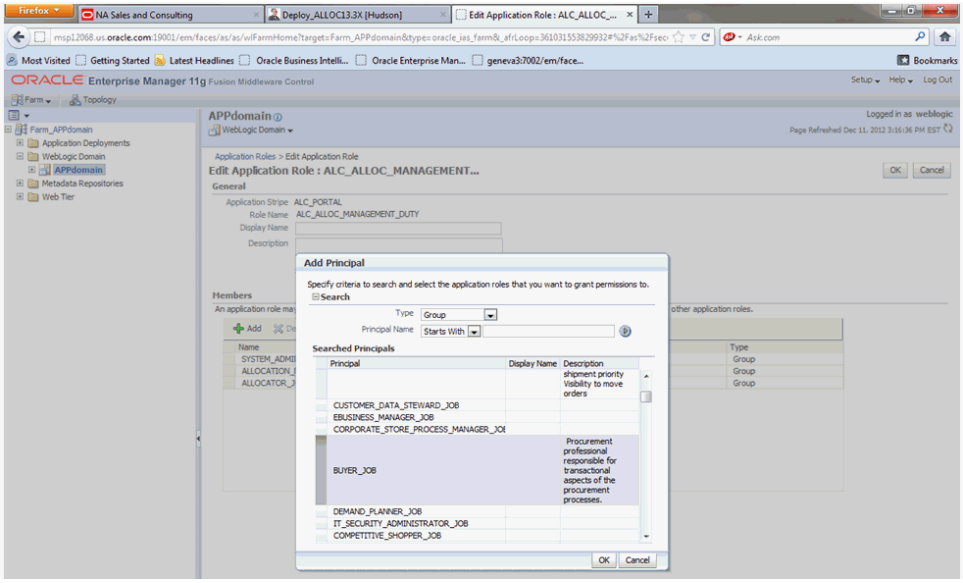
5. Select from the following options:

   - **To delete a member:** From **Members**, select from **Name** the member to activate the **Delete** button and click **Delete**.
   - **To add a member:** Click the **Add** button that corresponds to the member type being added. Select from **Add Application Role**, **Add Group**, and **Add User**.

6. For adding a member, complete **Search** and select from the available list and click **OK**.

   Figure 20–8 shows the **Add Group** dialog and after the **BUYER_JOB** group has been selected.

Figure 20–8 Add Group Dialog Window



The added member displays in the **Members** column corresponding to the application role modified in the **Application Roles** page.

Creating Application Roles Using Fusion Middleware Control

Following are the two methods for creating a new application role:

- **Create New**: A new application role is created. Members can be added at the same time or you can save the new role after naming it and add members later.
- **Copy Existing**: A new application role is created by copying an existing application role. The copy contains the same members as the original, and is made a Grantee of the same application policy. You can modify the copy as needed to finish creating the new role.

To create a new application role

1. Log in to Fusion Middleware Control, navigate to **Security**, then select **Application Roles** to display the **Application Roles** page.

   For information, see Accessing Oracle Enterprise Manager Fusion Middleware Control section.

2. Choose **Select Application Stripe to Search**, and then click the search icon next to **Role Name**.

   The Oracle Retail Fusion Application's application roles displays.

3. Click **Create** to display the **Create Application Role** page. You can enter all information at once or you can enter a **Role Name**, save it, and complete the remaining fields later.

4. In the **General** section, specify the following:

   - **Role Name** - Enter the name of the application role.
   - **Display Name** - Enter the display name for the application role. This is an optional field.
   - **Description** - Enter a description for the application role.This is an optional field.

5. In the **Members** section, select the groups, or application roles to be mapped to the application role.

6. Select **Add Application Role** or **Add Group** accordingly.

7. To search in the dialog box that displays, specify the following:

   - Enter a name in **Name** field and click the blue button to search.
   - Select from the results returned in the **Available** box.
   - Click **OK** to return to the **Create Application Role** page.
   - Repeat the steps until all members are added to the application role.

8. Click **OK** to return to the **Application Roles** page.

   The application role just created displays in the table at the bottom of the page.

To create an application role based on an existing one

1. Log in to Fusion Middleware Control, navigate to **Security**, then select **Application Roles** to display the **Application Roles** page.

For more information, see Accessing Oracle Enterprise Manager Fusion Middleware Control section.

2. Choose **Select Application Stripe to Search**, and then click the search icon next to **Role Name**.

   The Oracle Retail Fusion Application's application roles is displayed.

3. Select an application role from the list to enable the action buttons.

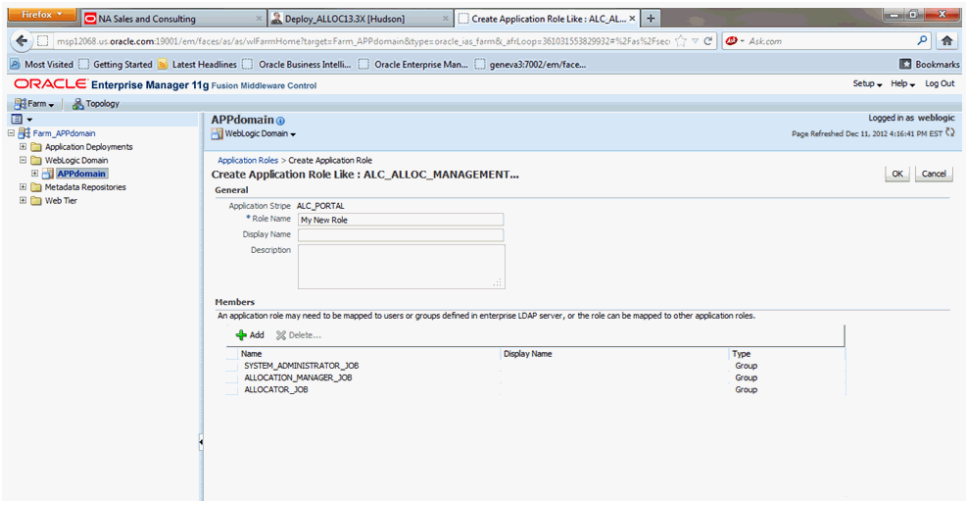
4. Click **Create Like** to display the **Create Application Role Like** page.

   The **Members** section is completed with the same application roles, groups that are mapped to the original role.

5. Complete the **Role Name**, **Display Name**, and **Description** fields.

   Example 20–9 shows an application role based upon ALC_ALLOC_MANAGEMENT_DUTY after being named **MyNewRole**, as an example.

Figure 20–9 Create Application Role Window



6. Use **Add** and **Delete** to modify the members as appropriate and click **OK**.

   The just created application role displays in the table at the bottom of the page. The following figure shows the example **MyNewRole** that is based upon the default ALC_ALLOC_MANAGEMENT_DUTY application role.

Customizing the Default Security Configuration

You can customize the default security configuration in the following ways:

- Create new application roles. For more information, see To create a new application role section.
- Modifying membership in an Application Role. For more information, see Modifying Application Roles Using Fusion Middleware Control section.

Customizing the Policy Store

The Fusion Middleware Security model can be customized for your environment by creating your own application roles and modifying membership of application roles. Existing application roles can be modified by adding or removing members as needed.

For more information about managing application policies and application roles, see *Oracle Fusion Middleware Application Security Guide*.

Note: Before creating a new application role and adding it to the default Oracle Retail Fusion Application's security configuration, familiarize yourself with how permission and group inheritance works. It is important when constructing a role hierarchy that circular dependencies are not introduced. The best practice is to leave the default security configuration in place and first incorporate your customized application roles in a test environment. For more information, see Permission Grants and Inheritance section.

Part VI

Active Retail Intelligence (ARI)

The following chapters provide guidance for administrators, developers, and system integrators who securely administer, customize, and integrate the Active Retail Intelligence (ARI) application.

Part VI contains the following chapter:

21

Security Considerations for Active Retail Intelligence (ARI)

This chapter covers the possible Simple Mail Transfer Protocol (SMTP) injections that may occur and possible workaround, though the customer is at liberty to implement any other measures based on industry best practices.

Active Retail Intelligence (ARI) provides no special security features or safeguards. Addressing any site-specific security issues involving ARI is the customer's responsibility. Security settings in other applications with which ARI interacts will not be overridden or circumvented by ARI. Whereas this is generally desirable, it is a consideration when determining to whom ARI alerts should be routed. Sending an alert to a user who does not have the privileges to take the actions necessary to resolve the event may prove frustrating and counter-productive. Users should be educated about this issue so that they can avoid forward events that have actions with limited access as well.

At a data level, ARI detection is necessarily done with full access privileges to all data. Individual users with data level security may see different values for some parameters (in particular those involving sums) than the values seen by ARI. This may cause adverse effects such as a user looking at an event automatically causing it to close because the user's limited data access causes the event to see values that make ARI think the exception is no longer an issue when in fact it still is. For this reason Oracle urges extreme caution when designing ARI processes that involve users with limited data access. The consequences of missing alerts can be great in an exception driven enterprise, so extra care is needed in the technical analysis of how such ARI processes will behave.

Simple Mail Transfer (SMTP) Injections

An attacker exploits the weakness in input validation on Internet Message Access Protocol (IMAP)/Simple Mail Transfer Protocol (SMTP) servers to execute commands on the server. Web-mail servers often sit between the Internet and the IMAP or SMTP mail server. User requests are received by the Web-mail servers which then query the back-end mail server for the requested information and return this response to the user. In an IMAP/SMTP command injection attack, mail-server commands are embedded in parts of the request sent to the Web-mail server. If the Web-mail server fails to adequately sanitize these requests, these commands are then sent to the back-end mail server when it is queried by the Web-mail server, where the commands are then executed. This attack can be especially dangerous since administrators may assume that the back-end server is protected against direct Internet access and therefore may not secure it adequately against the execution of malicious commands.

It is the customer's responsibility to sanitize the requests and ensure that the e-mail address is validated.

Following is one way to hack the e-mail alert sent by ARI:

- Having message body containing a line with a single dot '.' in it. This signifies the end of the current message. This enables the hacker to specify another message, including a set of SMTP headers and message body. To be secure, in places where just a single dot on an empty line is found, the single dot is removed. Hence, even if the message body contains a single dot, the email would not be hacked.