Oracle® Retail Warehouse Management System

Security Guide

Release 14.0

E50568-01

October 2013

Oracle Retail Warehouse Management System, Release 14.0

E50568-01

Primary Author: Philip Wells

Contributing Author:

Contents

Preface

This guide serves as a guide for administrators, developers, and system integrators who securely administer, customize, and integrate Oracle Retail Warehouse Management System applications. Installation and configuration for each product are covered in more detail in 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 in the Oracle Retail Warehouse Management System 14.0 documentation set:

- Oracle Retail Warehouse Management System Data Model
- Oracle Retail Warehouse Management System Installation Guide
- Oracle Retail Warehouse Management System Online Help

- Oracle Retail Warehouse Management System Operations Guide
- Oracle Retail Warehouse Management System Radio Frequency User Guide
- Oracle Retail Warehouse Management System Release Notes
- Oracle Retail Warehouse Management System User Interface User Guide
- Oracle Retail Integration Bus documentation

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 or additional patches, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch 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/technetwork/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-installing of Retail Infrastructure in WebLogic

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-installing steps for secured setup of Retail infrastructure in WebLogic.

The following topics are covered in this chapter:

Pre-installing - Steps for Secured Setup of 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 focus on securing 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 issues 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

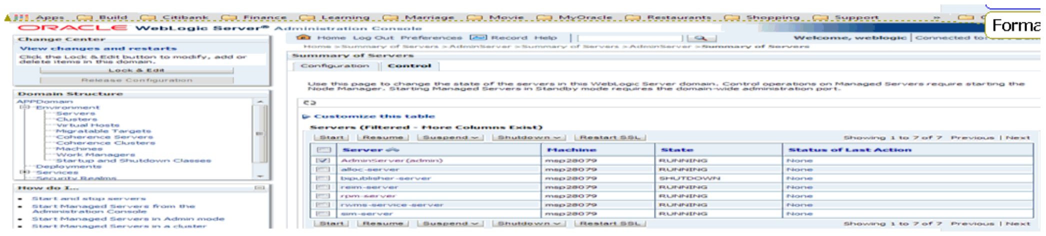
Creating a WebLogic Domain

WebLogic domain is created for the Retail Application 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 the 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



Click the Lock & Edit button to modify, add or delete items in this domain.

Home >Summary of Servers >AdminServer >Summary of Servers >AdminServer >Summary of Servers

Summary of Servers

Use this page to change the state of the servers in this WebLogic Server domain. Control operations on Managed Servers require starting the Node Manager. Starting Managed Servers in Standby mode requires the domain-wide administration port.

Servers (Filtered - More Columns Exist)

	Server	Machine	State	Status of Last Action
✓	AdminServer (admin)	msp28079	RUNNING	None
	alloc-server	msp28079	RUNNING	None
	bipublisher-server	msp28079	SHUTDOWN	None
	reim-server	msp28079	RUNNING	None
	rpm-server	msp28079	RUNNING	None
	rwms-service-server	msp28079	RUNNING	None
	sim-server	msp28079	RUNNING	None

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.

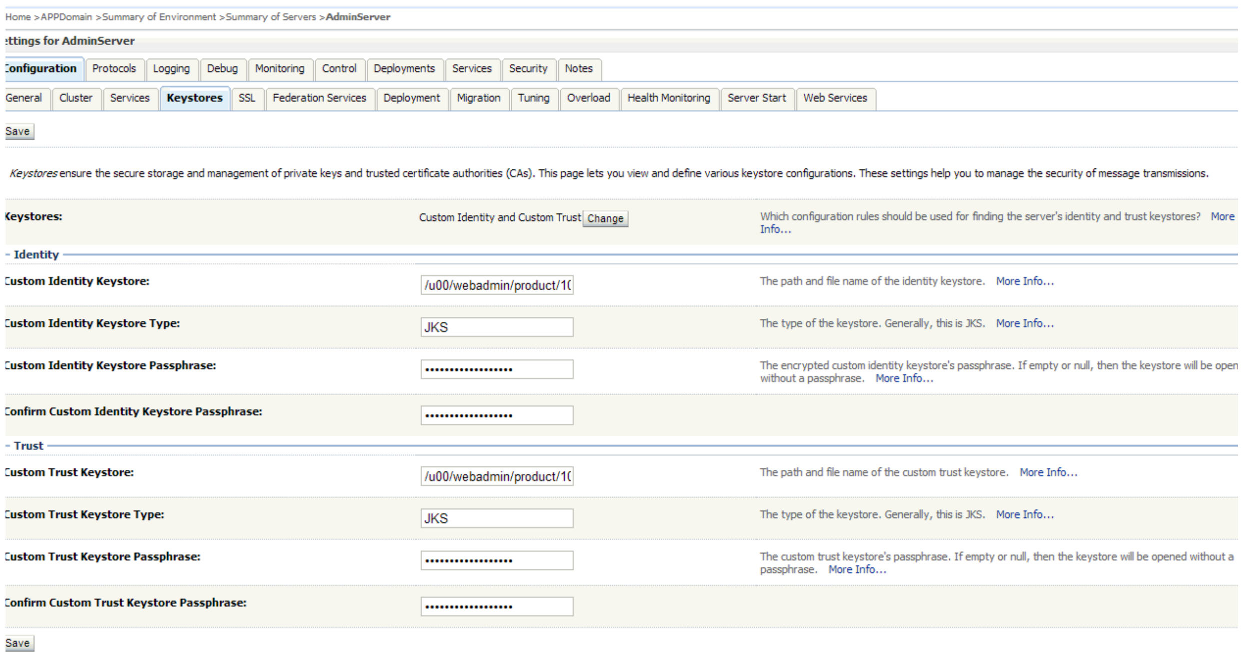
- **Custom Trust Keystore Passphrase** - This is the password that you need to enter when reading or writing to the keystore. This attribute is optional or required depending on the type of keystore. All keystores require the passphrase in order to write to the keystore. However, some keystores do not 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



Home >APPDomain >Summary of Environment >Summary of Servers >**AdminServer**

ttings for AdminServer

Keystores ensure the secure storage and management of private keys and trusted certificate authorities (CAs). This page lets you view and define various keystore configurations. These settings help you to manage the security of message transmissions.

eystores:	Custom Identity and Custom Trust Change	Which configuration rules should be used for finding the server's identity and trust keystores? More Info...
– Identity		
ustom Identity Keystore:	/u00/webadmin/product/1(	The path and file name of the identity keystore. More Info...
ustom Identity Keystore Type:	JKS	The type of the keystore. Generally, this is JKS. More Info...
ustom Identity Keystore Passphrase:	••••••••••••••••••	The encrypted custom identity keystore's passphrase. If empty or null, then the keystore will be open without a passphrase. More Info...
onfirm Custom Identity Keystore Passphrase:	••••••••••••••••••	
– Trust		
ustom Trust Keystore:	/u00/webadmin/product/1(	The path and file name of the custom trust keystore. More Info...
ustom Trust Keystore Type:	JKS	The type of the keystore. Generally, this is JKS. More Info...
ustom Trust Keystore Passphrase:	••••••••••••••••••	The custom trust keystore's passphrase. If empty or null, then the keystore will be opened without a passphrase. More Info...
onfirm Custom Trust Keystore Passphrase:	••••••••••••••••••	

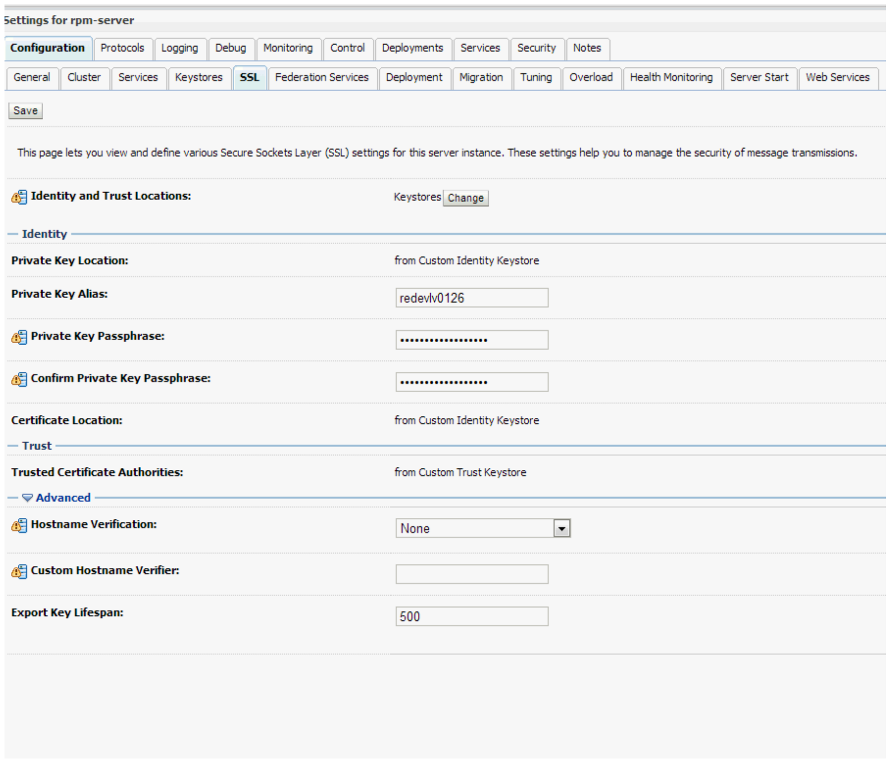
For more information on configuring Keystores, see the section *Configure Keystores* in the *Administration Console Online Help*.

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



This page lets you view and define various Secure Sockets Layer (SSL) settings for this server instance. These settings help you to manage the security of message transmissions.

Identity and Trust Locations:	Keystores
Identity	
Private Key Location:	from Custom Identity Keystore
Private Key Alias:	redevlv0126
Private Key Passphrase:	••••••••••••••••••
Confirm Private Key Passphrase:	••••••••••••••••••
Certificate Location:	from Custom Identity Keystore
Trust	
Trusted Certificate Authorities:	from Custom Trust Keystore
Advanced	
Hostname Verification:	None
Custom Hostname Verifier:	
Export Key Lifespan:	500

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 in case Admin Server is secured

Perform the following steps to configure the WebLogic scripts in case Admin Server to be secured:

There should be instructions for insuring the

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

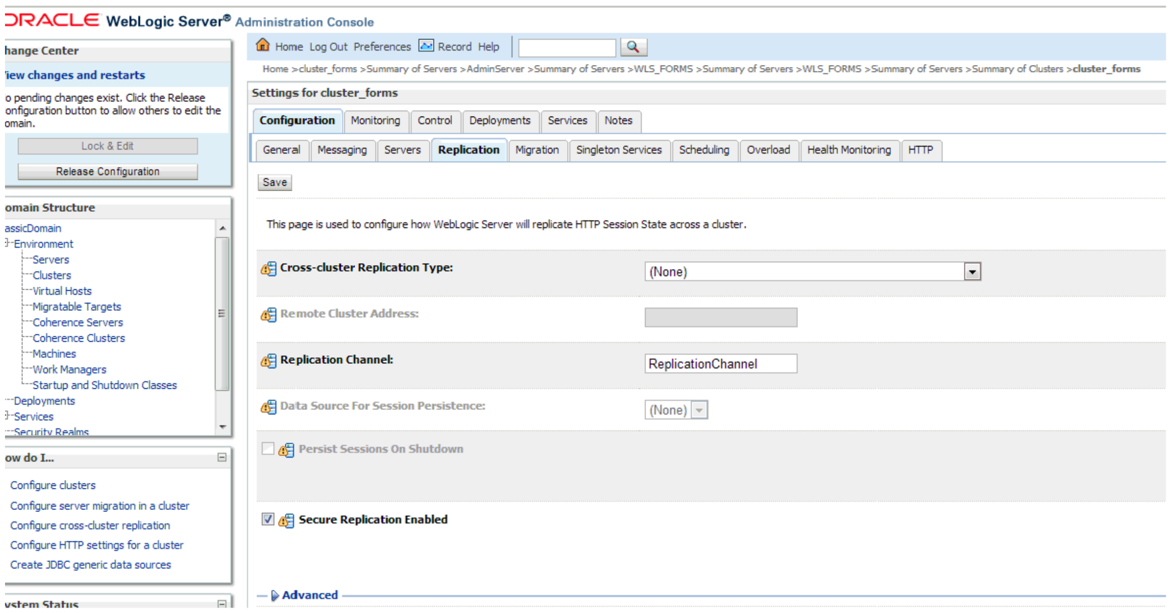
   https://apphost1:7002

Additional configuration for WLS_FORMS (For forms server)

Perform the following steps for WebLogic forms:

1. Log in to the **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



o pending changes exist. Click the Release onfiguration button to allow others to edit the omain.

Home >cluster_forms >Summary of Servers >AdminServer >Summary of Servers >WLS_FORMS >Summary of Servers >WLS_FORMS >Summary of Servers >Summary of Clusters >**cluster_forms**

Settings for cluster_forms

This page is used to configure how WebLogic Server will replicate HTTP Session State across a cluster.

Cross-cluster Replication Type:	(None)
Remote Cluster Address:	
Replication Channel:	ReplicationChannel
Data Source For Session Persistence:	(None)
☐ Persist Sessions On Shutdown	
☑ Secure Replication Enabled	

Adding certificate to the JDK keystore for installer

You will need the Retail Application installer to run Java. In situation 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

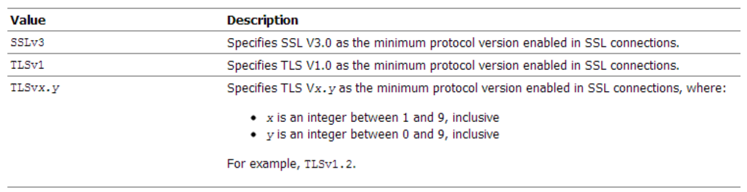
In a production environment, Oracle recommends Transport Layer Security (TLS) Version 1.1, or higher for sending and receiving messages in an SSL connection.

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]

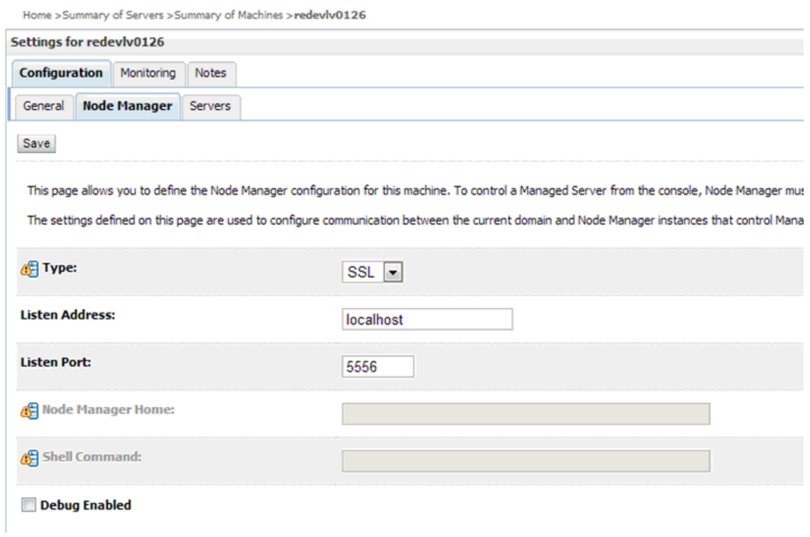
   CustomIdentityAlias=hostname

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

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

   SecureListener=true
3. Log in to the **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


Home >Summary of Servers >Summary of Machines >redevlv0126

Settings for redevlv0126

This page allows you to define the Node Manager configuration for this machine. To control a Managed Server from the console, Node Manager mus

The settings defined on this page are used to configure communication between the current domain and Node Manager instances that control Mana

Type:	SSL
Listen Address:	localhost
Listen Port:	5556
Node Manager Home:	
Shell Command:	

☐ Debug Enabled

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 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 Retail Service Bus (RSB) security policies

Perform the additional pre-requisites for Retail Service Bus (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 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
- Configuring the Password Stores for Database User Accounts
- Configuring the Database Password Policies
- Additional Information

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

The *Oracle Database Security Guide* covers more information on the subjects covered in this section as well as information on other options that are available to strengthen database security.

For more information, 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 Retail infrastructure in WebLogic.

The following topics are covered in this chapter:

- Batch set up for SSL Communication
- Oracle Business Intelligence (BI) Publisher - Disable guest user - Optional
- RWMS - Timeout Setting (Optional)

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, Retail Price Management (RPM), Store Inventory Management (SIM), and Retail Invoice Matching (ReIM). 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.

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**..

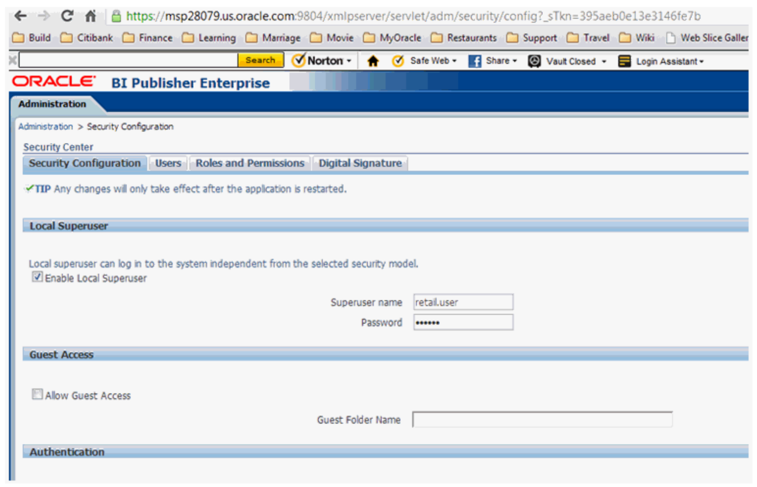
Oracle Business Intelligence (BI) Publisher - Disable guest user - Optional

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 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



https://msp28079.us.oracle.com:9804/xmlpserver/servlet/adm/security/config?_sTkn=395aeb0e13e3146fe7b

ORACLE BI Publisher Enterprise

Administration

Administration > Security Configuration

Security Center

Security Configuration | Users | Roles and Permissions | Digital Signature

✓TIP Any changes will only take effect after the application is restarted.

Local Superuser

Local superuser can log in to the system independent from the selected security model.

[✓] Enable Local Superuser

Superuser name: retail.user

Password: ••••••

Guest Access

[ ] Allow Guest Access

Guest Folder Name:

Authentication

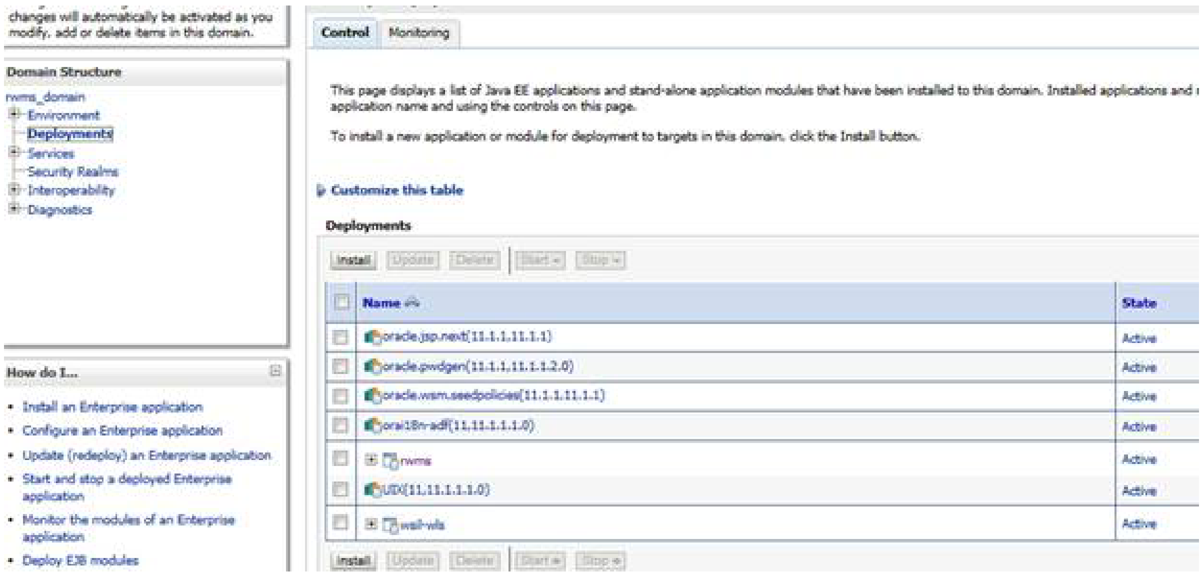
4. Save and restart the BI Publisher instance.

RWMS - Timeout Setting (Optional)

The RWMS Application comprises of ADF and Forms components and hence is deployed in two Weblogic domains. Due to this fact, the timeout of RWMS is always controlled from the ADF Application. Accordingly, the timeout should be set in the ADF Domain.

Select the RWMS Application in Deployments.

Figure 3–2 RWMS - Deployments


changes will automatically be activated as you modify, add or delete items in this domain.

Domain Structure

rwms_domain
- Environment
- Deployments
- Services
- Security Realms
- Interoperability
- Diagnostics

How do I...

- Install an Enterprise application
- Configure an Enterprise application
- Update (redeploy) an Enterprise application
- Start and stop a deployed Enterprise application
- Monitor the modules of an Enterprise application
- Deploy EJB modules

Control | Monitoring

This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and
application name and using the controls on this page.

To install a new application or module for deployment to targets in this domain, click the Install button.

Customize this table

Deployments

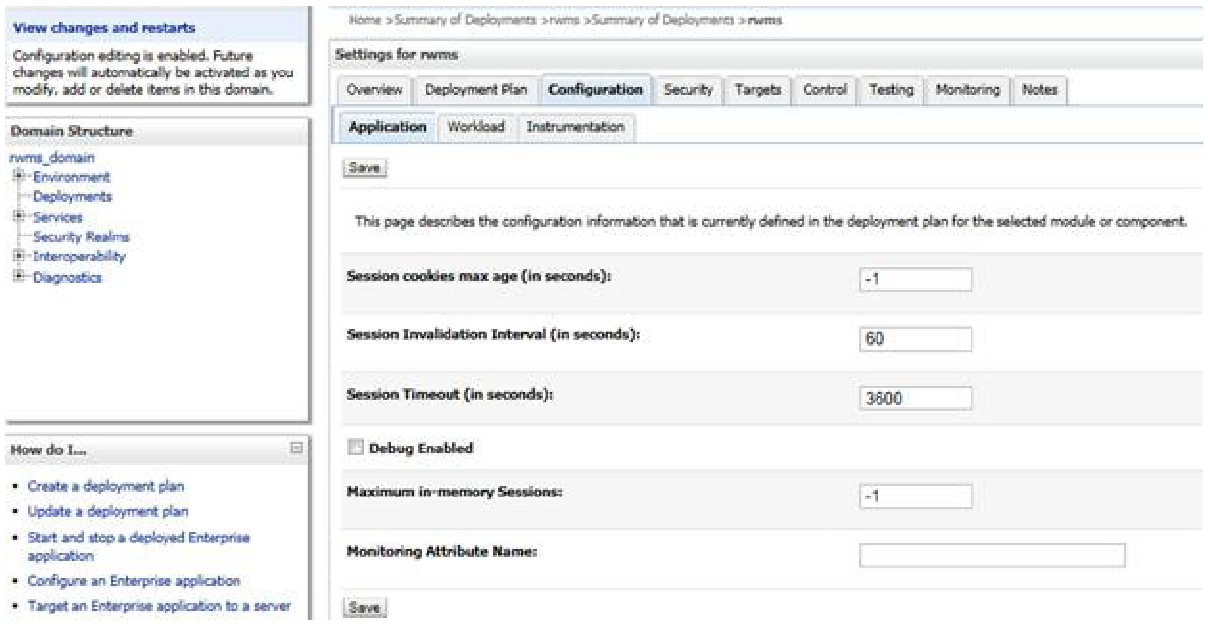
Install | Update | Delete | Start | Stop

	Name	State
☐	oracle.jsp.next(11.1.1,11.1.1)	Active
☐	oracle.pwdgen(11.1.1,11.1.1.2.0)	Active
☐	oracle.wsm.seedpolicies(11.1.1,11.1.1)	Active
☐	orai18n-adf(11,11.1.1.1.0)	Active
☐	rwms	Active
☐	UIX(11,11.1.1.1.0)	Active
☐	wsil-wls	Active

Install | Update | Delete | Start | Stop

Navigate to the Configuration tab.

Figure 3–3 Deployments - Configuration Tab


View changes and restarts

Configuration editing is enabled. Future changes will automatically be activated as you modify, add or delete items in this domain.

Domain Structure

rwms_domain
- Environment
- Deployments
- Services
- Security Realms
- Interoperability
- Diagnostics

How do I...

- Create a deployment plan
- Update a deployment plan
- Start and stop a deployed Enterprise application
- Configure an Enterprise application
- Target an Enterprise application to a server

Home >Summary of Deployments >rwms >Summary of Deployments >rwms

Settings for rwms

Overview | Deployment Plan | Configuration | Security | Targets | Control | Testing | Monitoring | Notes

Application | Workload | Instrumentation

Save

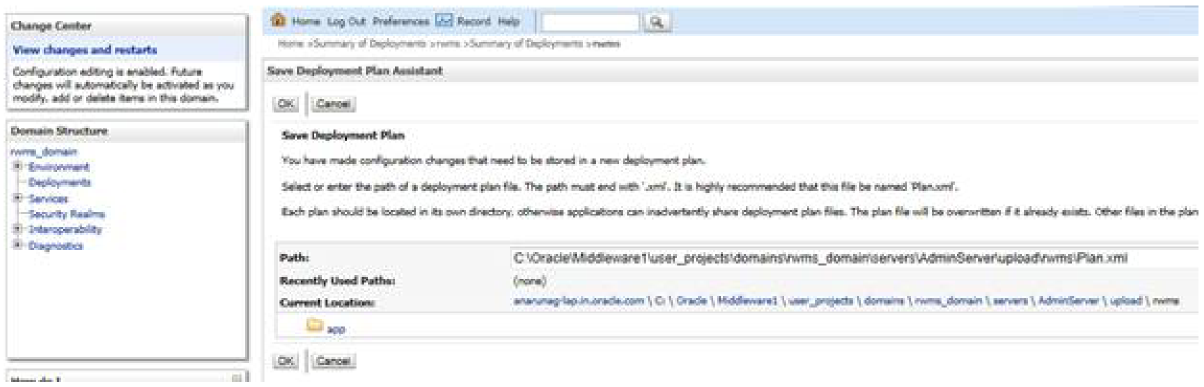
This page describes the configuration information that is currently defined in the deployment plan for the selected module or component.

Session cookies max age (in seconds):	-1
Session Invalidation Interval (in seconds):	60
Session Timeout (in seconds):	3600
☐ Debug Enabled	
Maximum in-memory Sessions:	-1
Monitoring Attribute Name:	

Save

Modify the Session Timeout (in seconds) to the necessary interval. Setting it to 0 will cause the application never to timeout. Save the changes.

Figure 3–4 Deployment - Save Changes



Configuration editing is enabled. Future changes will automatically be activated as you modify, add or delete items in this domain.

Home >Summary of Deployments >rwms >Summary of Deployments >rwms

Save Deployment Plan Assistant

Save Deployment Plan

You have made configuration changes that need to be stored in a new deployment plan.

Select or enter the path of a deployment plan file. The path must end with '.xml'. It is highly recommended that this file be named 'Plan.xml'.

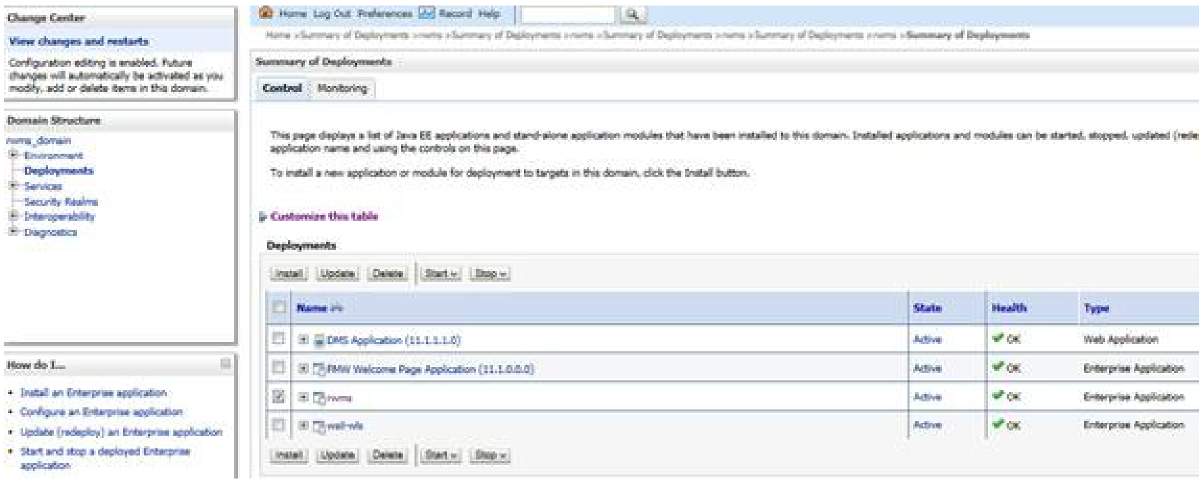
Each plan should be located in its own directory, otherwise applications can inadvertently share deployment plan files. The plan file will be overwritten if it already exists. Other files in the plan

Path:	C:\Oracle\Middleware1\user_projects\domains\rwms_domain\servers\AdminServer\upload\rwms\Plan.xml
Recently Used Paths:	(none)
Current Location:	anarunag-lap.in.oracle.com \ C: \ Oracle \ Middleware1 \ user_projects \ domains \ rwms_domain \ servers \ AdminServer \ upload \ rwms

Click Ok.

Select RWMS Application under Deployments and press the Update button.

Figure 3–5 Deployment - Update Option



Configuration editing is enabled. Future changes will automatically be activated as you modify, add or delete items in this domain.

- Install an Enterprise application
- Configure an Enterprise application
- Update (redeploy) an Enterprise application
- Start and stop a deployed Enterprise application

Home >Summary of Deployments >rwms >Summary of Deployments >rwms >Summary of Deployments >rwms >Summary of Deployments >rwms >**Summary of Deployments**

Summary of Deployments

This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and modules can be started, stopped, updated (rede application name and using the controls on this page.

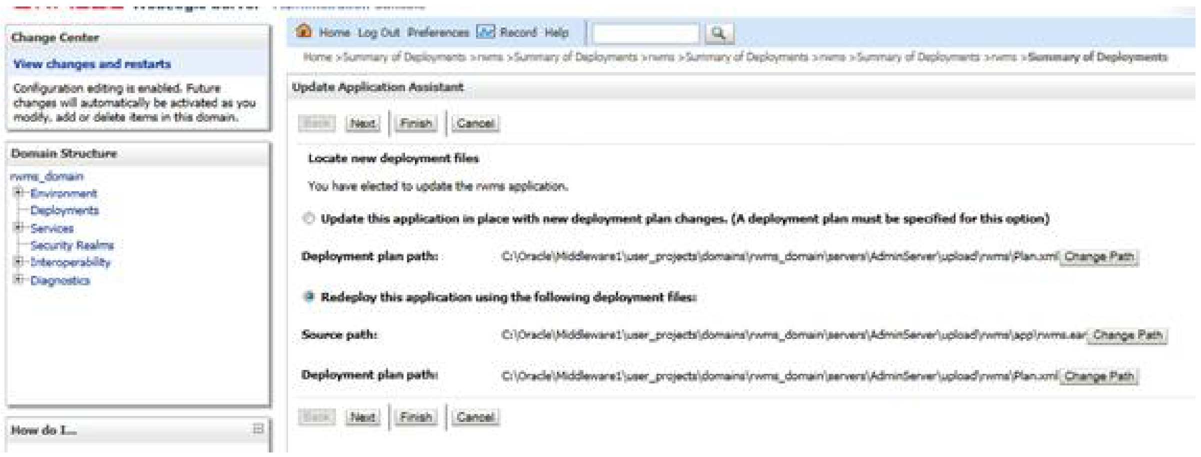
To install a new application or module for deployment to targets in this domain, click the Install button.

Deployments

	Name	State	Health	Type
☐	DMS Application (11.1.1.1.0)	Active	✓ OK	Web Application
☐	RMW Welcome Page Application (11.1.0.0.0)	Active	✓ OK	Enterprise Application
☑	rwms	Active	✓ OK	Enterprise Application
☐	wall-wls	Active	✓ OK	Enterprise Application

Click Finish.

Figure 3–6 Deployment - Finish Option



Configuration editing is enabled. Future changes will automatically be activated as you modify, add or delete items in this domain.

Home >Summary of Deployments >rwms >Summary of Deployments >rwms >Summary of Deployments >rwms >Summary of Deployments >rwms >**Summary of Deployments**

Update Application Assistant

Locate new deployment files

You have elected to update the rwms application.

○ **Update this application in place with new deployment plan changes. (A deployment plan must be specified for this option)**

Deployment plan path:	C:\Oracle\Middleware1\user_projects\domains\rwms_domain\servers\AdminServer\upload\rwms\Plan.xml

[✓ **Redeploy this application using the following deployment files:**]

Source path:	C:\Oracle\Middleware1\user_projects\domains\rwms_domain\servers\AdminServer\upload\rwms\app\rwms.ear
Deployment plan path:	C:\Oracle\Middleware1\user_projects\domains\rwms_domain\servers\AdminServer\upload\rwms\Plan.xml

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

You need to set the following parameters:

FORMS_TIMEOUT

This parameter is set using RMS/RWMSenvfilecreatedat<DOMAIN_ HOME>/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_ 11.1.2/configdirectory

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.

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

HeartBeat

This parameter is set in the formsweb.cfg file located in the 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(1day).

Example 3–2 HeartBeat Configuration

```
[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
```

For RWMS it is mandatory that the value of HeartBeat is less than that for FORMS_ TIMEOUT. It is recommended that the default values be left unchanged.

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

4 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

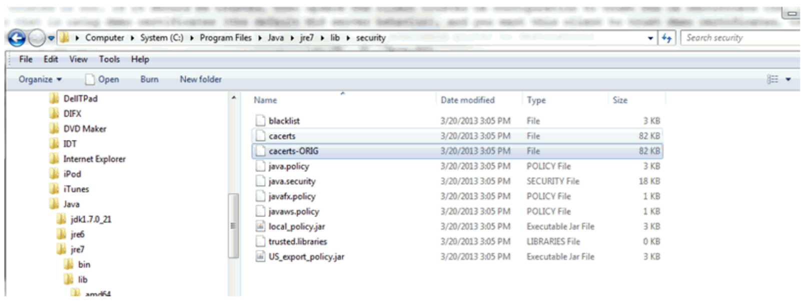
If customers are using certificates other than provided by standard certificate authorities like custom CA implementation, then the JRE used for launching the applications from local machines like laptops or desktops might display a different error messages.

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 4–1 Cacert backup


Computer ▸ System (C:) ▸ Program Files ▸ Java ▸ jre7 ▸ lib ▸ security
Search security
File Edit View Tools Help
Organize Open Burn New folder
DellTPad
DIFX
DVD Maker
IDT
Internet Explorer
iPod
iTunes
Java
jdk1.7.0_21
jre6
jre7
bin
lib
amd64

Name	Date modified	Type	Size
blacklist	3/20/2013 3:05 PM	File	3 KB
cacerts	3/20/2013 3:05 PM	File	82 KB
cacerts-ORIG	3/20/2013 3:05 PM	File	82 KB
java.policy	3/20/2013 3:05 PM	POLICY File	3 KB
java.security	3/20/2013 3:05 PM	SECURITY File	18 KB
javafx.policy	3/20/2013 3:05 PM	POLICY File	1 KB
javaws.policy	3/20/2013 3:05 PM	POLICY File	1 KB
local_policy.jar	3/20/2013 3:05 PM	Executable Jar File	3 KB
trusted.libraries	3/20/2013 3:05 PM	LIBRARIES File	0 KB
US_export_policy.jar	3/20/2013 3:05 PM	Executable Jar File	3 KB

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.

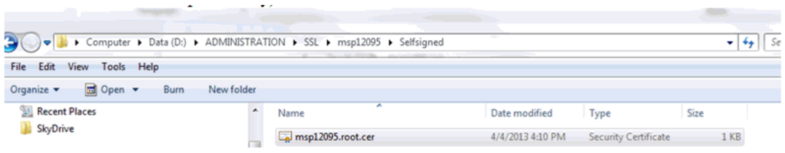
Importing the Root Certificate through Internet Explorer

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 4–2 Importing the Root Certificate file to the workstation


Computer ▸ Data (D:) ▸ ADMINISTRATION ▸ SSL ▸ msp12095 ▸ Selfsigned
File Edit View Tools Help
Organize Open Burn New folder
Recent Places
SkyDrive

Name	Date modified	Type	Size
msp12095.root.cer	4/4/2013 4:10 PM	Security Certificate	1 KB

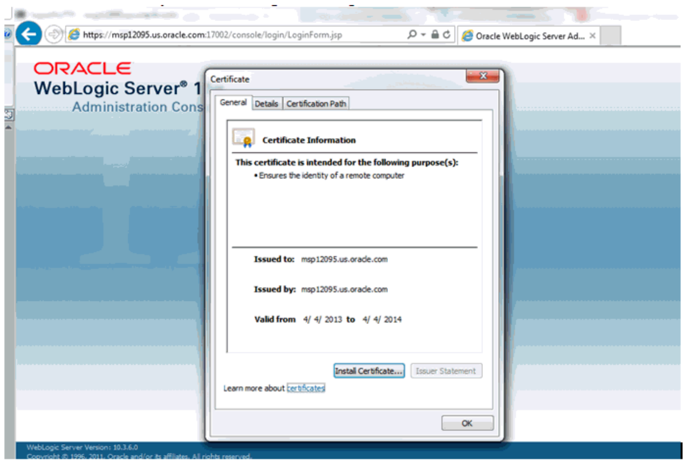
3. Select the file.
4. Click **Install Certificate** and click **Next**.
5. Select **Place all certificates in the following store** and click **Browse**.
6. Select **trusted Root Certification Authorities** and click **OK**.
7. Click **Next**.
8. Click **Finish** and then **Yes** at the Security Warning 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 4–3 Importing the Root Certificate file through Mozilla Firefox



Certificate Information

This certificate is intended for the following purpose(s):
• Ensures the identity of a remote computer

Issued to: msp12095.us.oracle.com

Issued by: msp12095.us.oracle.com

Valid from 4/ 4/ 2013 to 4/ 4/ 2014

Learn more about certificates

WebLogic Server Version: 10.3.6.0
Copyright © 1996, 2011, Oracle and/or its affiliates. All rights reserved.

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:
EC:A7:16:DF:B6

Signature algorithm name: SHA1withRSA

Version: 3

Extensions:

Verifying Keystore Content

Keystores are repository of the certificates. In situations when we are facing issues related to SSL Certificates, one can check the certificates which are available in the keystore. In case the certificates are not missing, they should be imported. 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

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 in "Pre-installing of Retail Infrastructure in WebLogic" of Chapter 1.

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 we 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 4–1, "WLS_Forms startup error". To resolve, ensure that Additional configuration for WLS_FORMS (For forms server) in Pre-installing of Retail Infrastructure in WebLogic chapter have been completed. The startup shows the errors in the logs as shown in the example, when we try to restart the WLS_FORMS managed server in Oracle Forms installation after configuring for security.

Example 4–1   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**.

5 Importing Topology Certificate

Implementation of SSL into the 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 Retail Applications

Table 5–1 describes the trust stores to be updated while confirming the certificates imported into middleware and repository of 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 5–1, the *root.cer are the public key certificates and the *server.cer are the private key certificates.

Table 5–1 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

6 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 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

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 in Chapter 1.

Configuring Nodemanager

You need to secure the Node manager by following the steps in Securing Nodemanager with SSL Certificates section in Chapter 1.

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 in Troubleshooting chapter.

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 Warehouse Merchandising System

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

Part II contains the following chapters:

- RWMS Architecture
- RWMS Administration

Additional information may also be found in the I*nstallation Guide for the Oracle Retail Warehouse Management System*.

7

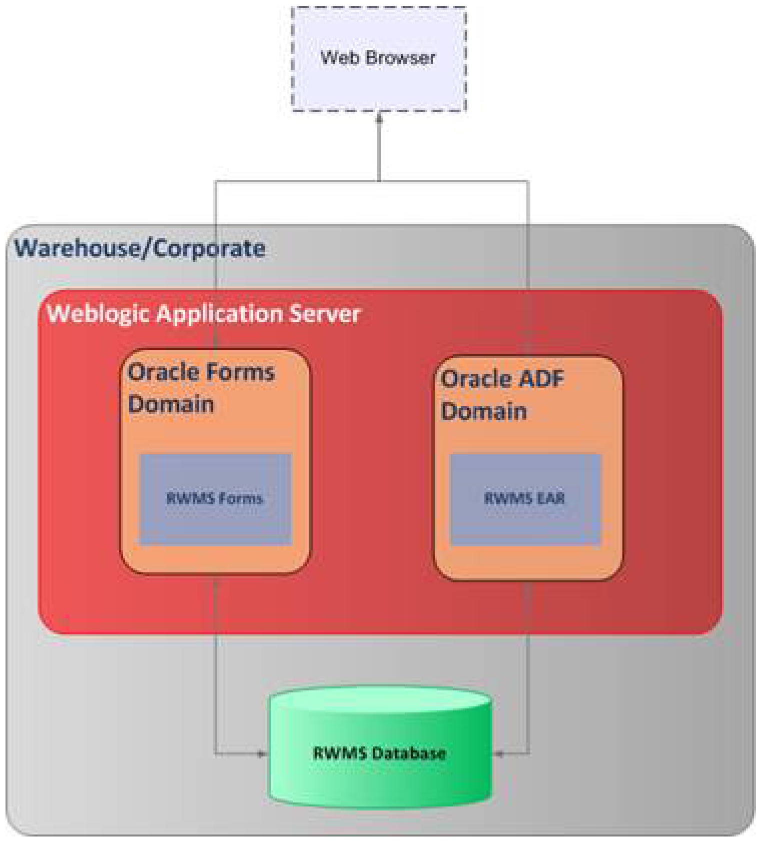
RWMS Architecture

This chapter discusses security related aspects of RWMS architecture.

Architecture

RWMS 14.0 has been developed using both Oracle ADF and Oracle Forms with integration between both components. This requires that the deployment of RWMS involves two instances of a WebLogic Application Server; one hosting the ADF Application and the other hosting the Oracle Forms Application.

Figure 7–1 RWMS Architecture



The RWMS application will use the following mechanisms to secure the application:

- A standard JAAS based application server based authentication for user identification and setting access roles.
- An ADF based JAZN security model to map enterprise roles obtained from the previous step to application roles defined within the application to provide another layer of abstraction.

Authentication

The new architecture necessitates the secure passing of user credentials from ADF Application to Forms Application. A new table, **RWMS_USER_SESSIONS**, has been created for this purpose. The ADF Application will insert a valid row in this table after a successful login. Forms Application will verify the contents of this table and after successful validation will allow the application to continue. After validation, the Forms Application will delete the corresponding row in the table.

To ensure that there are no orphaned records in the **RWMS_USER_SESSIONS** table (which may provide valuable information to attackers) a database job **RWMS_PURGE_SECURITY_OBJ** runs every 2 minutes to remove the records which are more that 120 seconds old.

Hardening

The frequency of the database job **RWMS_PURGE_SECURITY_OBJ** can be changed by running the shell script **rwms_alter_purge_sec_obj_sch.sh**. The parameter value is in minutes. For example, a value of 1 can be passed as the parameter to change the frequency of the job to 60 seconds.

Setting the frequency to a lower value could be an additional burden on the database and would need to be evaluated in the production scenario. It is suggested that the interval be not more than 60 minutes to ensure that all the sensitive information is cleared.

Timeout

Since RWMS is now deployed on two WebLogic Instances (ADF and Forms), timeout of the application needs to be tightly coupled between these two servers. Timeout for the Oracle Forms Application will not be set, so that the ADF Application will invalidate both the sessions and exit the application.

If the Forms Application is active, it will send a keep-alive request to the ADF Application to keep the ADF Application active as well. If the timeout of the ADF Application is 60 minutes, the keep-alive request will be sent every 15 minutes.

To ensure that the keep-alive request is not sent from malicious applications, a record is inserted into the **XDOMAIN_MESSAGES** table before the keep-alive request is sent. The ADF Application will then validate the inserted record in the **XDOMAIN_MESSAGES** table and upon successful validation keep itself alive. If the validation fails then the ADF Application will insert a message in **XDOMAIN_QUEU**E. This will be browsed by the corresponding Forms Application and the Forms Application will then close itself. The ADF Application will also invalidate the session.

Hardening

Hardening may be carried out by doing the following:

- Set the timeout of the ADF Application to a reduced interval. The default timeout is 35 Minutes.

- The frequency of the database job **RWMS_PURGE_SECURITY_OBJ** can be changed by running the shell script **rwms_alter_purge_sec_obj_sch.sh**. An increased purge rate will improve security but may slow the application.

Tables in Encrypted Tablespace

All the tables that store sensitive information (including PII data and Application User Credentials) are created in an encrypted tablespace. This is to ensure that the printable strings in the datafiles are hidden away from attackers. Following is the list of tables which are stored in encrypted tablespace.

ACTIVITY_LOG
AHL_QUERIES
APPLY_CONTAINER_ROUTE
APPOINTMENT
APPT_LINE_WORKING
BOL_HEADER
BOL_HEADER_TO_UPLOAD
CARRIER
CONT_LABELS_TO_PRINT
CONTAINER
CONTAINER_HISTORY
CONTAINER_ITEM_ATTRIBUTE
CONTAINER_ROUTE
CONTAINER_ITEM_ATTRIBUTE
CONTAINER_ROUTE
CONTAINER_WIP
COUNTRY_CODES
CYCLE_COUNT_ADJUSTMENTS
CYCLE_COUNT_LOG
DC_VIEW_METRICS
DIST_CONTAINER
DIST_PICK_DIRECTIVE
DISTRIBUTION_QUEUE
DISTRIBUTION_ROUTE
DMS_FAVORITES_MENU
DMS_USER
ERROR_LOG
EXPLODE_KITS
EXTERNAL_MSG
FIND_DEFAULT
FPR_CONFIRMATION
FPR_CONTAINER
GENERIC_TEMP
GET_CASE_LABEL
INV_ADJUSTMENT_TO_UPLOAD
INV_ADJUSTMENT_TO_UPLOAD_HIS
KIT_ASSEMBLE_LIST
LOCATION
MANUAL_ORDER_QUERIES
NON_CONFORM_ATTACHMENTS
NON_CONFORM_ATTACHMENTS
NON_CONFORM_DETAIL
NON_CONFORM_ENTRIES
ORDER_QUERIES
ORDER_TOTALS

OUTBOUND_QC
OUTSTANDING_TOTALS
OVERAGES_TO_UPLOAD
PATCHES_INSTALLED
PEND_CONTAINER_LIST
PICK_DIRECTIVE
PTS_QUERIES
QC_AUDIT
RECEIPT_TO_UPLOAD
RECEIPT_TO_UPLOAD_HIS
RECEIVING_OVERRIDES
RECV_ADJ_LOG
RECV_PKG_TO_PRINT
REPLEN_DIRECTIVE
RMA_SEARCH_QUERIES
ROUTE
RTV
SELECT_WIP
SELECTED_ACTIVITIES
SELECTED_APPTS
SELECTED_DESTS
SELECTED_DISTROS
SHIP_DEST
SHIP_DEST_ORDER_CUBE
SPACE_UTILIZATION
STANDING_APPOINTMENT
STOCK_ALLOCATION
STOCK_ORDER
STORE_WT_CUBE
TASK_LOG
TASK_LOG_DETAILS
TASKS
TMP_CONTAINER_ITEM
TRANSPORT_INVENTORY
UNIT_PICK_GROUP
UPS_CHUTE_DETAIL
USER_ACTIVITY_GROUPS
USER_ATTRIBUTE
USER_EQUIPMENT
USER_EXCEPTION
USER_SHIFTS
USER_TICKET_TYPES
VENDOR_ADDRESS
VENDOR_TROUBLE_HISTORY
ZONE_TO_WAVE

Separate Run Time User

A new database runtime user has been introduced to prevent breach of privileges. This should be used to login to RWMS Application. This user will have only the required privileges to run the application. This user or wallet alias for this user should be the value provided for **userid** in formsweb.cfg file.

```
rwms132F]
envfile=./develop/rwms132F.env
width=100%
```

```
height=100%
separateFrame=true
form=logon_scr.fmx
lookAndFeel=Oracle
colorScheme=swan
archive=frmall.jar,icons.jar
imageBase=codebase
userid=wms01user/password@connect_string
or
userid=walletalias
```

8 RWMS Administration

This section covers how to improve security when dealing with the administration of RWMS.

Administration

This section covers security aspects of administration.

Password Administration

The following Secure Copy Parameters (SCPs) should be used for Password Administration. The usage description is provided in the section SCP Parameters for Security below.

- hashing_algorithm
- min_password_length
- min_password_length
- Password_expire

Password Encryption

User passwords will be stored using the hashing algorithms provided by the Oracle database. The hashing algorithm which will be used can be changed by modifying the SCP **hashing_algorithm**.

The default value of the SCP is **DBMS_CRYPTO.HASH_SH1**. Other allowed values are **DBMS_CRYPTO.HASH_MD4** and **DBMS_CRYPTO.HASH_MD5**.

If the SCP value is modified, the passwords will be hashed using the new algorithm when the user changes his password.

Authorization - User Privileges

At install time and when utilizing the RWMS copy facility functionality, RWMS populates the **dms_menu** table with all of the menu options that make up the RWMS application.

In the **dms_menu** table there is a column called **user_privilege**. This column is used in conjunction with the **user_privilege** column within the **dms_user** table to restrict users from accessing menu options from the RWMS Main Menu screens. It also prevents users from being able to access a screen via the task administration functionality where the application directs users from screen to screen to complete tasks without navigating through menu options.

Each screen can be configured via the Menu Editor to set it's **user_privilege** level. When a user logs into the RWMS application the **dms_user.user_privilege** setting is evaluated against the **dms_menu.user_privilege setting** to determine what menu options should be available.

The configuration of the **user_privilege** setting should be based on business needs. It is recommended that users carrying out administrative tasks should have higher privileges than users carrying out day to day tasks.

Note: When the **dms_menu** table is populated at installation time via the base install scripts the **user_privilege** level is always set to 1. When the **dms_menu** table is populated via the copy facility functionality, the **user_privilege** is copied over as well. These default values will probably require modification to improve security.

Hardening

Depending on the business needs, all the menu options need to be assigned the desired privilege so that the users get access to only the necessary forms.

SCP Parameters for Security

The following SCP parameters will help in strengthening the security of the application:

- **hashing_algorithm**: This should be set to the hashing algorithm provided by Oracle Database. The default value is DBMS_CRYPTO.HASH_SH1.
- **min_password_length**: Default length of password is 7. For hardening, this can be increased up to 20.
- **max_invld_login_cnt**: The default value is 5 and indicates the number of invalid login attempts allowed. For hardening this value should be reduced.
- **ac_cycle_count_priv**: The default value is 7. This specifies the privilege level that allows users to perform Audit Counts.For hardening, this should be increased to reduce the number of users with the those privileges. The maximum privilege level is 9.
- **password_complexity**: The default value is ANX. The options are:
  - Set to N for numeric only passwords.
  - Set to A for alphabetic only passwords.
  - Set to AN for Alphanumeric only passwords (One alphabet and one number mandatory).
  - Set to ANX for Alphanumeric and any other special character based password. (Minimum of one alphabet, one number and one special character mandatory).
  - Set to X for any character based password.

  ANX is the suggested and the strongest setting. Any other setting will leave the system prone to brute force attacks.
- **password_expire**: The number of days after which password expires. The default value is 365. For hardening this value should be reduced.

Batch Jobs

File Permissions

The Shell Scripts should normally not have Write Permissions for Group and Other users. Permissions should be set to 755 or lower. For hardening ensure that the user accessing the batch jobs are part of group and do not provide access to other users

Oracle Connection

The shell scripts will run the SQL statements using the RWMS Owning schema. Ensure that the appropriate wallet alias is created for this user and run the profile for executing Shell Scripts as documented in the Operations Guide.

RIB Integration

RIB should use the **Separate Runtime User** to connect to RWMS. Refer to the *Oracle Retail Integration Bus* documentation for more information on how to set up the integration.