Oracle® Fusion Middleware
Administrator’s Guide for Oracle API Catalog
12c Release 1 (12.1.3)
E55977-03

October 2015
Documentation for administrators that describes infrastructure, lifecycle, and security administration tasks for Oracle API Catalog
Contents

Preface................................................................................................................................................................ v
Audience......................................................................................................................................................... vii
Documentation Accessibility .................................................................................................................. vii
Related Documents ............................................................................................................................. vii
Conventions .............................................................................................................................................. vii

1 Introducing Administration Tasks in Oracle API Catalog
  1.1 Introducing Infrastructure Administration Tasks ................................................................. 1-1
  1.2 Introducing Asset Lifecycle Administration Tasks ............................................................ 1-2
  1.3 Introducing Security Administration Tasks ......................................................................... 1-2

2 Basic Configuration
  2.1 Overview ....................................................................................................................................... 2-1
  2.2 Departments ............................................................................................................................... 2-1
  2.2.1 Creating a New Department ............................................................................................... 2-1
  2.2.2 Editing Department Information ....................................................................................... 2-2
  2.2.3 Deleting a Department ........................................................................................................ 2-2
  2.3 Roles ........................................................................................................................................... 2-2
  2.4 Users ........................................................................................................................................... 2-3
  2.4.1 Creating a New User ............................................................................................................. 2-3
  2.4.2 Viewing User Information ................................................................................................. 2-4
  2.4.3 User Search .......................................................................................................................... 2-4
  2.4.4 Editing User Information .................................................................................................... 2-4
  2.4.5 Deleting a User .................................................................................................................... 2-5
  2.5 Sessions ....................................................................................................................................... 2-5
  2.5.1 Viewing Sessions ............................................................................................................... 2-5
  2.5.2 Deleting a Single Session ................................................................................................. 2-5
  2.5.3 Deleting Multiple Sessions .............................................................................................. 2-5
  2.6 Accessing Oracle API Catalog Diagnostics Page .................................................................. 2-6
  2.7 Install Java Web Start ................................................................................................................ 2-6
  2.7.1 Download and Install Java Web Start ............................................................................... 2-6
  2.7.2 Troubleshooting Java Web Start ....................................................................................... 2-6
  2.7.3 Configure Java Web Start on Linux .................................................................................... 2-7
3 System Settings Overview

3.1 Access System Settings ................................................................. 3-1
3.2 Search for System Settings ........................................................... 3-1
3.3 Move Settings to Database ............................................................. 3-1
3.4 System Settings in Oracle API Catalog ........................................ 3-2
3.4.1 Functional Settings ................................................................. 3-2
3.4.1.1 Search ............................................................................. 3-2
3.4.1.2 Print Asset Detail via PDF ............................................... 3-2
3.4.2 Server Settings ...................................................................... 3-3
3.4.2.1 General .......................................................................... 3-3
3.4.3 Enterprise Authentication Settings ........................................... 3-3
3.4.3.1 General .......................................................................... 3-3
3.4.3.2 Plugin Login Settings ....................................................... 3-4
3.4.3.3 LDAP and Active Directory Settings .................................. 3-4
3.4.3.4 Container Login Settings ................................................ 3-6
3.4.4 Import / Export and Introspection Settings ............................... 3-7
3.4.4.1 Import / Export ............................................................... 3-7

4 Import Export Tool

4.1 Overview ................................................................................... 4-1
4.2 Using the Import/Export Tool ..................................................... 4-2
4.2.1 Overview ................................................................................ 4-2
4.2.2 The parameter.properties File .............................................. 4-2
4.2.2.1 Supported Parameters in the parameter.properties File ........ 4-3
4.2.2.2 Sample parameter.properties File ..................................... 4-5
4.3 Starting the Import/Export Tool .................................................. 4-6
4.3.1 Launching from Oracle API Catalog ....................................... 4-6
4.3.2 Launching from Outside of Oracle API Catalog ...................... 4-6
4.3.3 Initial Startup ........................................................................ 4-6
4.4 Export Items from Oracle API Catalog ....................................... 4-7
4.5 Importing Items into Oracle API Catalog .................................... 4-7

5 Configuring and Using Automated Harvesting

5.1 Getting Started with Harvester ................................................... 5-1
5.1.1 Prerequisites ......................................................................... 5-1
5.1.2 Harvester Functionality .......................................................... 5-2
5.1.3 Artifacts/Products Version Matrix .......................................... 5-2
5.2 Configuring the Harvester ........................................................... 5-2
5.2.1 Obtaining the Harvester .......................................................... 5-3
5.2.2 Configuring the Harvester for the Command Line .................. 5-3
5.2.2.1 Setting Repository Connection Information for the Command Line ........................................................................................................... 5-4
5.2.2.2 Selecting the Artifacts to Harvest for the Command Line ........ 5-4
5.2.3 Invoking the Harvester Using the Repository.Submit Ant Task ... 5-7
5.2.3.1 Specifying Parameters for the repository.submit Ant Task .... 5-7
5.2.3.2 Specifying Parameters as Nested Elements ....................... 5-8
5.2.3.3 Running the Harvester from Ant ........................................ 5-10
Preface

Oracle Fusion Middleware Administrator’s Guide for Oracle API Catalog provides the information needed to complete the three types of administration tasks in Oracle API Catalog: infrastructure administration, asset lifecycle administration, and security administration.

Audience

This document is intended for individuals who are administering an Oracle API Catalog installation or curating the API asset lifecycle process.

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 that have purchased support 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 API Catalog 12c documentation set:

- Oracle Fusion Middleware Concepts Guide for Oracle API Catalog
- Oracle Fusion Middleware Installation Guide for Oracle API Catalog
- Oracle Fusion Middleware Developer’s Guide for API Catalog

Conventions

The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>boldface</td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td>Convention</td>
<td>Meaning</td>
</tr>
<tr>
<td>------------</td>
<td>---------</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td><code>monospace</code></td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
This part presents introductory information about administering Oracle API Catalog. This part includes the following chapters:

- Chapter 1, "Introducing Administration Tasks in Oracle API Catalog"
This chapter describes the types of administration tasks in Oracle API Catalog. Administration of Oracle API Catalog falls into three areas: setting up and maintaining the Catalog infrastructure, managing the asset lifecycle, and administering security.

This chapter contains the following sections:

- Section 1.1, "Introducing Infrastructure Administration Tasks"
- Section 1.2, "Introducing Asset Lifecycle Administration Tasks"
- Section 1.3, "Introducing Security Administration Tasks"

### 1.1 Introducing Infrastructure Administration Tasks

Maintaining the repository infrastructure includes product setup and configuration for areas such as user authentication, departments and system settings. These tasks are typically performed during the initial setup of OAC before it is populated with APIs, with periodic updates as needed.

The first task, which is optional in OAC, is to setup Departments. Departments can be used to group users organizationally; they can reflect departments or groups within the organization and can also represent groups, such as partners or other third parties, that need access to OAC. OAC includes one out of the box department, named "Default Department". Users are not required to be assigned to departments; creating departments is optional.

For more information on setting up departments, see Section 2.2, "Departments".

The next task is to setup user authentication so users can access OAC. During the user setup process, users are assigned one of the three roles included with OAC, which determines which OAC capabilities the user has access to. See the "Understanding User Roles in Oracle API Catalog" section in the Oracle Fusion Middleware Concepts Guide for Oracle API Catalog for more information about the roles available in OAC.

There are 2 options for creating users:

- Integrate with an existing LDAP implementation. See Chapter 7, "Configuring Oracle API Catalog to use External Authentication Tooling" for more information about configuring OAC to work with external authentication.
- Use administration capabilities to create users in OAC. See Section 2.4.1, "Creating a New User" for more information about creating users.
You can optionally perform the other administrative tasks in Part II, "Administering the Infrastructure" if your deployment requires them.

After these tasks are completed, the tasks described in Section 1.2, "Introducing Asset Lifecycle Administration Tasks" should be completed.

1.2 Introducing Asset Lifecycle Administration Tasks

Managing the asset lifecycle is focused on the creation, maintenance, and use of the assets in Oracle API Catalog.

The first activity in the asset lifecycle is to harvest APIs into OAC. The curator uses the harvester to populate OAC with API assets from SOA Suite and Service Bus or other deployed services. The harvester can be run from the command line or be integrated into the build process to automatically do the harvest at build time. The harvester creates API assets in a "Draft" state, meaning they are not discoverable by developers searching OAC. See Chapter 5, "Configuring and Using Automated Harvesting" for more information about harvesting APIs.

After harvesting, the API assets are edited by the curator using a simple editor. To facilitate the discovery and understanding of the API, metadata such as an updated name, complete and meaningful description, keywords, and a documentation reference should be added to the API. See Section 6.1, "Editing the Metadata of an API Asset" for information about editing API metadata.

After the API metadata is edited and complete, curator publishes the API, making it visible to developers in OAC. See Section 6.2, "Publishing an API Asset" for information about publishing APIs.

Published APIs are visible in JDeveloper and the OAC Console for discovery by developers. Developers can search OAC to discover APIs they may want to use, they can examine the API metadata to learn more about an API, and they can add the API their personal "My APIs" list where they can provide ratings and reviews for the API. For more information on how a user with the developer role uses OAC, see the Oracle Fusion Middleware Developer's Guide for API Catalog.

At the end of their life an API is "Retired" by the curator using the pop-up API editor. After the API is retired it will not be visible to Developers when they search OAC. An API is retired by selecting Retired from the Active Status list, as described in Section 6.1, "Editing the Metadata of an API Asset".

1.3 Introducing Security Administration Tasks

Administering Security in Oracle API Catalog is primarily focused on controlling access to the content using Authentication and Authorization.

OAC uses the two following methods for authentication:

- OAC integrates with the organization’s LDAP server. For more information, see Chapter 7, "Configuring Oracle API Catalog to use External Authentication Tooling".

- Organizations can also create users directly in OAC. Created users is discussed in Section 2.4.1, "Creating a New User".

Users are assigned one of the OAC predefined roles: admin, curator, or developer. These roles determine which OAC capabilities the user is authorized to use. See the "Understanding User Roles in Oracle API Catalog" section in the Oracle Fusion Middleware Concepts Guide for Oracle API Catalog for more information about the roles available in OAC.
Part II
Administering the Infrastructure

This part describes basic administration tasks for the Oracle API Catalog infrastructure.

This part includes the following chapters:

- Chapter 2, "Basic Configuration"
- Chapter 3, "System Settings Overview"
- Chapter 4, "Import Export Tool"
This chapter describes the basics of how to configure Oracle API Catalog by introducing you to the terminology and common tasks of an Oracle API Catalog administrator.

This chapter contains the following sections:

- Section 2.1, "Overview"
- Section 2.2, "Departments"
- Section 2.3, "Roles"
- Section 2.4, "Users"
- Section 2.5, "Sessions"
- Section 2.6, "Accessing Oracle API Catalog Diagnostics Page"
- Section 2.7, "Install Java Web Start"

2.1 Overview

In Oracle Enterprise Repository, all users are assigned roles. Roles define the Oracle Enterprise Repository functionality that can be exercised by each user. Users are assigned to Departments. This is convenient from a reporting standpoint, as organizations can then track the production and consumption of reusable assets to a specific department.

2.2 Departments

You need to set up or create departments before creating users and projects. This section contains the following topics:

- Section 2.2.1, "Creating a New Department"
- Section 2.2.2, "Editing Department Information"
- Section 2.2.3, "Deleting a Department"

2.2.1 Creating a New Department

To create a new department, perform the following steps from the Oracle API Catalog Admin screen:

1. In the left panel, click Departments.
2. Click the **Create New Department** icon, as shown in Figure 2–1.

![Create New Department Icon](image.png)

This image displays a portion of the Oracle Enterprise Repository Admin screen. The User Menu is displayed at the top of the image. The User Menu are the API Catalog Home, My APIs, and Admin buttons are displayed below the User Menu. The Create New Department icon is displayed at the bottom of the image. The mouse pointer is hovering over the Create New Department icon.

3. Enter the appropriate information into the **Name** and **Description** fields.

4. When finished, click **Save**.

### 2.2.2 Editing Department Information

To edit a department information, perform the following steps from the Oracle API Catalog Admin screen:

1. In the left panel, click **Departments**.

2. Select the department to be edited from the list in the right panel. The department's detail page is displayed.

3. Click the **Edit** icon. The Edit Department dialog is displayed.

4. Update the **Name** and **Description** fields as necessary.

5. When finished, click **Save**.

### 2.2.3 Deleting a Department

To delete a department in Oracle API Catalog, perform the following steps from the Oracle API Catalog Admin Screen:

1. In the left panel, click **Departments**.

2. To delete a department:

   - To delete departments from the Search Results screen, select the check box next to each department that you want to delete, and then click **Delete**. You can delete multiple departments simultaneously using this method.

   - To delete a department from its detail page, open the detail page for the department you want to delete, and then click **Delete**.

### 2.3 Roles

Oracle Enterprise Repository access settings are established for each role, and each user is assigned a specific role based on his/her function within the organization.
Roles cannot be added, edited, or removed in Oracle Enterprise Repository; users can be assigned only the default roles. For a description of the default roles see "Role-Based Features" in the Oracle Fusion Middleware Concepts Guide for Oracle Enterprise Repository.

Roles can be changed on a per-user basis. See Section 2.4.4, "Editing User Information" to edit a user’s role.

2.4 Users

This section describes the following topics:

- Section 2.4.1, "Creating a New User"
- Section 2.4.2, "Viewing User Information"
- Section 2.4.3, "User Search"
- Section 2.4.4, "Editing User Information"
- Section 2.4.5, "Deleting a User"

2.4.1 Creating a New User

To create a new user, perform the following steps on the Oracle API Catalog Admin screen:

1. In the left panel, click Users.

2. Click the Create New User icon, as shown in Figure 2–2. The Create New User page is displayed.

![Create New User Icon](image)

This image displays a portion of the Oracle Enterprise Repository Admin screen. The User Menu is displayed at the top of the image. The User Menu are the API Catalog Home, My APIs, and Admin buttons are displayed below the User Menu. The Create New User icon is displayed at the bottom of the image. The mouse pointer is hovering over the Create New User icon.

3. Enter the appropriate information in each of the text fields in the Overview section, as shown in Figure 2–3.

---

**Note:** Ensure that you supply values for each of the required fields, denoted by an asterisk.
This figure illustrates the Overview section.

4. Select **Active** in the Status list.

5. In the Roles section, assign roles to the new user by using the arrow buttons to move items from the **Available Roles** column to the **Selected Roles** column.

6. In the Departments section, assign the new user to departments by using the arrow buttons to move items from the **Available Departments** column to the **Selected Departments** column.

7. When finished, click **Save**.

### 2.4.2 Viewing User Information

To view a user's information, perform the following steps on the Oracle API Catalog Admin screen:

1. In the left panel, click **Users**.

2. Use **Search** to locate the user(s) to be viewed.

3. Select a user from the list. The user's information is displayed in a new tab.

### 2.4.3 User Search

This procedure is performed on the Oracle API Catalog Admin screen.

1. In the left panel, click **Users**.

2. Enter appropriate text in the **Name** field.

3. Use the **Department**, **Role**, and **Status** lists as necessary to narrow the search.

4. Click the **Search** button. Search results appear in the list in the main pane.

### 2.4.4 Editing User Information

This procedure is performed on the Oracle API Catalog Admin screen.

1. In the left panel, click **Users**.
2. Use **Search** to locate the user(s) to be edited. The list of users appears in the main pane.

3. Select a user from the list. The user’s information is displayed in a new tab.

4. Click the **Edit** icon. The Edit User dialog is displayed.

5. Make changes as necessary in the **Overview**, **Roles**, and **Departments** sections.

6. When finished, click **Save**.

### 2.4.5 Deleting a User

To delete a user in Oracle API Catalog, perform the following steps from the Oracle API Catalog Admin Screen:

1. In the left panel, click **Users**.

2. Use **Search** to locate the user(s) to be deleted. The list of users appears in the main pane.

3. To delete a user:
   - To delete users from the Search Results screen, select the check box next to each user that you want to delete, and then click **Delete**. You can delete multiple users simultaneously using this method.
   - To delete a user from its detail page, open the detail page for the user you want to delete, and then click **Delete**.

### 2.5 Sessions

Sessions allow the administrator to see which users are logged in to Oracle API Catalog and, if necessary, shut down a user session.

- **Section 2.5.1, "Viewing Sessions"**
- **Section 2.5.2, "Deleting a Single Session"**
- **Section 2.5.3, "Deleting Multiple Sessions"**

#### 2.5.1 Viewing Sessions

This procedure is performed on the Oracle API Catalog Admin screen.

1. In the left panel, click **Sessions**. A list of sessions is displayed in the main pane.

#### 2.5.2 Deleting a Single Session

1. Select the session to be deleted from the list in the main pane. The session’s detail opens in a new tab.

2. Click the **Delete** button in the session detail. A confirmation dialog is displayed.

3. Click **OK**. The session is deleted.

#### 2.5.3 Deleting Multiple Sessions

1. In the list of sessions in the main pane, select each session to be deleted.

   **Tip:** Place a check mark in the check box at the top of the column to select all listed sessions.
2. Click the **Delete** button in the table header. A confirmation dialog is displayed.

3. Click **OK**. The session is deleted.

### 2.6 Accessing Oracle API Catalog Diagnostics Page

In Oracle API Catalog, the Diagnostics page is disabled, by default. Navigate to http://host_name:port/application_name/diag/index.jsp (replace **host_name** with the appropriate location).

When you open the Diagnostics page in the default mode, the following message is displayed:

*Diag pages are currently disabled. Please contact your Oracle API Catalog Administrator.*

To enable the Diagnostics page, perform the following steps:

1. Navigate to `ORACLE_HOME/user_projects/domains/DOMAIN_NAME/bin/`.
2. Edit `setStartupEnv.sh` or `setStartupEnv.cmd` before starting the Oracle API Catalog server.
3. Add `-DdiagPagesEnabled=true` to the `EXTRA_JAVA_PROPERITES="${EXTRA_JAVA_PROPERTIES}"` line in this file.
4. Restart the Oracle API Catalog server.

It is recommended to only enable when necessary and disable once the system is running without any issues.

### 2.7 Install Java Web Start

Java Web Start is a browser plug-in that runs files with JNLP extensions. Java Web Start must be available on the client to use Oracle API Catalog’s Import/Export utility.

- **Section 2.7.1, "Download and Install Java Web Start"
- **Section 2.7.2, "Troubleshooting Java Web Start"
- **Section 2.7.3, "Configure Java Web Start on Linux"

#### 2.7.1 Download and Install Java Web Start

Java Web Start is included in the latest version of the JDK.

At the time this document was published, Oracle Fusion Middleware products required JDK 7.0 Update 55 or later. However, you should always verify the required JDK version by reviewing the certification information, as described in the documentation. To download the required JDK, use your browser to navigate to the following URL and download the Java SE JDK:


#### 2.7.2 Troubleshooting Java Web Start

When browsers encounter a file with an unknown extension, they often prompt you to Open or Save when opening an application. Choose the **Open** option and browse to the `$JAVA_HOME/jre/bin/javaws` file, and select the appropriate option, to perform this action for the JNLP file type. If a new JRE is required, then follow the instructions in **Section 2.7.1, "Download and Install Java Web Start"** to download the latest JRE.
2.7.3 Configure Java Web Start on Linux

You must configure Java WebStart before you can use it on a Linux operating system.

To configure Java WebStart:

1. Close Mozilla Firefox.
2. Locate the mozilla directory on your computer, for example, /usr/lib/mozilla-1.4 or /home/<user>/.mozilla.
3. Open a command prompt and go to the mozilla directory on your computer, using the following command, for example:
   ```
   cd home/<user>/.mozilla
   ```
4. Create the plugins directory, if it does not exist, using the following command:
   ```
   mkdir plugins
   ```
5. Go to the plugins sub-directory under the Mozilla installation directory, as follows:
   ```
   cd plugins
   ```
6. In the current directory, create a symbolic link to the Java ns7/libjavaplugin_oji.so file type:
   ```
   ln -s /oracle/soa/mwhome_gov/jdk7/jre/plugin/i386/ns7/libjavaplugin_oji.so
   ```
7. Open Mozilla Firefox and test the following link:
   ```
   ```
8. In Mozilla Firefox, click the Edit menu, and then select Preferences. The Preferences dialog is displayed.
9. Click Applications. The Application page is displayed.
10. In the Search field, enter JNLP. The JNLP is displayed.
11. Select the displayed JNLP. The Action list gets activated.
12. Click the Action list to point to the correct javaws installed (such as the one installed with WLS JDK's).

**Note:** However, on 64 bit Linux, WLS comes with JRockit only. You have to manually download the 64 bit JDK to access the plug-in.

**Note:** You can also run a Java Web Start application by saving the application locally, and then using the javaws executable, located in <JAVA_HOME>/jre/bin. For example:
```
<JAVA_HOME>/jre/bin/javaws -verbose /path_to_file/impexp.jnlp
```
This chapter describes Oracle API Catalog’s System Settings that administrators use to configure basic operations and specific features.

This chapter contains the following sections:

- Section 3.1, "Access System Settings"
- Section 3.2, "Search for System Settings"
- Section 3.3, "Move Settings to Database"
- Section 3.4, "System Settings in Oracle API Catalog"

### 3.1 Access System Settings

To access the system settings, perform the following steps:

1. Click the **Admin** button to open the Oracle API Catalog Admin page.
2. Click **System Settings** in the sidebar on Admin page.
   
   The System Settings section is displayed. System Settings are organized into categories. Each category is further divided into feature groups.
3. Click the plus icon to expand the display of any category or feature group.
4. Click the minus icon to collapse the display.

### 3.2 Search for System Settings

System settings for particular features can be located using the System Settings Search.

1. Enter a search term in the **Search** field. The relevant category sections expand to display the relevant System Settings.
2. Click **Cancel** to clear the search results.

### 3.3 Move Settings to Database

**Note:** Exercising this option moves all system settings stored in property files to the database. This action cannot be reversed.

Click **Move settings to database**. All system settings are saved to the database. This is for a clustered environment. All servers in the clustered environment will now have one source for system settings.
3.4 System Settings in Oracle API Catalog

This section describes all the system setting available in Oracle API Catalog.

This section contains the following topics:

■ Section 3.4.1, "Functional Settings"
■ Section 3.4.2, "Server Settings"
■ Section 3.4.3, "Enterprise Authentication Settings"
■ Section 3.4.4, "Import / Export and Introspection Settings"

3.4.1 Functional Settings

This section contains the following topics:

■ Section 3.4.1.1, "Search"
■ Section 3.4.1.2, "Print Asset Detail via PDF"

3.4.1.1 Search

Properties

■ Maximum number of results for Asset Search
  – cmee.search.assets.maxresults
  – The numerical value entered in the text box determines the maximum number of assets returned by a search. Entering 0 returns an unrestricted number.

■ Search Specific Field
  – cmee.search.specific.field
  – The True and False options toggle the ability to search specific fields during Standard Searches by identifying the field to be searched in the Keyword text box.

3.4.1.2 Print Asset Detail via PDF

Properties

■ Use default XSLT for asset types
  – cmee.asset.pdf.xslt.default
  – The True and False options specifies that when exporting an asset to PDF, a default asset type is used. Failing its existence, an error is shown.

■ PDF Asset Export Destination
  – cmee.asset.xml.paths.export-destination
  – Text entered in the text box identifies the PDF asset export destination.

■ PDF Asset Export Destination URL
  – cmee.asset.xml.paths.export-destination-url
  – Text entered in the text box identifies the PDF asset export destination URL.

■ PDF Asset XSL Directory
  – cmee.asset.xml.paths.xsl-source
Text entered in the text box identifies the PDF asset XSL directory.

- **PDF Asset Generated PDF Destination**
  - cmee.asset.xml.paths.out-destination
  - Text entered in the text box identifies the PDF asset generated PDF destination.

- **PDF Asset Generated PDF Destination URL**
  - cmee.asset.xml.paths.out-destination-url
  - Text entered in the text box identifies the PDF asset generated PDF destination URL.

- **PDF Asset Export WebApp URL**
  - cmee.asset.xml.paths.webapp-url
  - Text entered in the text box identifies the PDF asset export WebApp URL.

### 3.4.2 Server Settings
This section contains the following topics:

- **Section 3.4.2.1, "General"**

#### 3.4.2.1 General

- **Asset Editor Enable Embedded HTML in Asset Description**
  - cmee.asseteditor.enable-embedded-html-asset-description
  - True allows embedded HTML to be entered into an API's description.

### 3.4.3 Enterprise Authentication Settings
This section contains the following topics:

- **Section 3.4.3.1, "General"**
- **Section 3.4.3.2, "Plugin Login Settings"**
- **Section 3.4.3.3, "LDAP and Active Directory Settings"**
- **Section 3.4.3.4, "Container Login Settings"**

#### 3.4.3.1 General

**Properties**

- **Unapproved User Login**
  - enterprise.security.unapproveduser.allowlogin
  - The True and False options toggle the ability for unapproved users to log in to the system. This is useful when using LDAP or container-managed authentication.

- **Enable LDAP Login**
  - enterprise.authentication.ldap.enabled
  - Enables the LDAP Login Module for authentication.
3.4.3.2 Plugin Login Settings

**Properties**

- Plug-in Login Module Class Name
  - enterprise.loginmodules.pluggableloginmodule.classname
  - Text entered in the text box identifies the class name for the plug-in login module.
  
  Click +Add as necessary to add class names for additional plug-in login modules.

3.4.3.3 LDAP and Active Directory Settings

**Properties**

- LDAP Server Host Name
  - ldap.host
  - Enter the host name for the LDAP server. This is a mandatory field.

- LDAP Server Port Number
  - ldap.port
  - Enter the LDAP server port number. The default is 389. This is an optional field.

- LDAP Mask
  - ldap.mask
  - Enter the LDAP mask, either the authentication bind or search criteria. This is a mandatory field.

- Creation of Unapproved User Accounts
  - ldap.allow-user-creation
  - Select True to determine if unapproved user accounts can be created. The default value is true. This is an optional field.

- Assign default roles to users
  - ldap.assign-default-roles
  - Select true to determine, if users shall be assigned default roles.

- Auto create missing roles
  - ldap.auto-create-missing-roles
  - Select true to determine, if auto create missing roles is enabled.

- Auto create missing departments
  - ldap.auto-create-missing-depts
  - Select true to determine if auto create missing departments is enabled.

- LDAP Version
  - ldap.version
  - The LDAP version used, which can be either 2 or 3. The default value is 2. This is an optional field.
- Administrator Account Distinguished Name
  - ldap.bindDN
  - The distinguished name of the administrator account. This is an optional field.

- Administrator Account Password
  - ldap.bindPassword
  - The password for the administrator account. This is an optional field.

- Use SSL Connection
  - ldap.ssl.enabled
  - Enables an SSL Connection for LDAP. The default is false.

- Follow referrals
  - ldap.follow-referrals
  - If true, follow referrals.

- Retrieve data using the admin account
  - ldap.retrieve-data-as-admin
  - If true, then bindDN must be set. The default is false.

- Search Start Location
  - ldap.baseDN
  - Start location for a search, works with ldap.mask and ldap.scope to define the search. This is an optional field.

- Search Scope
  - ldap.scope
  - Determines the scope of the search. This is an optional field.

- Attribute Name that Identifies a Found Entry
  - ldap.uniqueIDAttrib
  - Attribute name that identifies a found entry beyond the use of the distinguished name, if not specified, then defaults to uid. This is a mandatory field.

- Found Entry Email Attribute Name
  - ldap.emailAttrib
  - Email attribute name of a found entry. This is an optional field.

- Found Entry First Name Attribute Name
  - ldap.givennameAttrib
  - First name attribute name of a found entry. This is an optional field.

- Found Entry Middle Name Attribute Name
  - ldap.middlenameAttrib
  - Middle name attribute name of a found entry. This is an optional field.

- Found Entry Last Name Attribute Name
  - ldap.surnameAttrib
- Last name attribute name of a found entry. This is an optional field.

- **Found Entry Telephone Number Attribute Name**
  - ldap.telephoneAttrib
  - Telephone number attribute name of a found entry. This is an optional field.

- **Use LDAP Departments**
  - ldap.enable-synch-depts
  - If true, get user departments from LDAP upon login.

- **Department Attribute**
  - ldap.deptAttrib
  - Name of attribute of the departments of the entry found.

- **Use LDAP Roles**
  - ldap.enable-synch-roles
  - If true, get user roles from LDAP upon login.

- **Role Attribute**
  - ldap.rbac.roleAttrib
  - Name of attribute of the roles of the entry found.

- **Second Level Lookup Attribute**
  - ldap.redirectDnAttrib
  - Attribute which identifies a second level lookup to retrieve user info; must be a dn(distinguished name). This is an optional field.

- **Bridge Class Name**
  - ldap.bridge.classname
  - This is a required field. The text for this property represents the class name of the bridge to be used.

### 3.4.3.4 Container Login Settings

**Properties**

- **Container Login Module Display Name**
  - enterprise.loginmodules.containerloginmodule.displayname
  - Text entered in the text box appears as the display name of the container login module.

- **Container Login Module**
  - enterprise.loginmodules.containerloginmodule.enabled
  - Select the True option to enable the container login module.

- **Container Login Module - Internal Checking**
  - enterprise.loginmodules.containerloginmodule.internalchecking
  - Select the True option to enable internal checking for the container login module.
3.4.4 Import / Export and Introspection Settings

This section contains the following topics:

- Section 3.4.4.1, "Import / Export"

3.4.4.1 Import / Export

Properties

- Import/Export job monitor max idle (msecs)
  - cmee.extframework.impexp.monitor.maxidle
  - The value entered in the text field represents the maximum time (in msecs) that an import/export job can be idle before it is automatically killed by the job monitor.

- Import/Export job monitor max idle for Rex transactions(msecs)
  - cmee.extframework.impexp.monitor.rex.maxidle
  - The value entered in the text field represents the maximum time (in msecs) that a rex transaction can be idle before it is automatically killed by the job monitor.

- Import/Export job monitor max runtime (msecs)
  - cmee.extframework.impexp.monitor.maxruntime
  - The value entered in the text field represents the maximum time (in msecs) that an import/export job can run before it is automatically killed by the job monitor.

- Import/Export job monitor period (msecs)
  - cmee.extframework.impexp.monitor.period
  - The value entered in the text field represents the time (in msecs) between scans for the job monitor.

- Import/Export job proxy period (msecs)
  - cmee.extframework.impexp.proxy.period
  - The value entered in the text field represents the time (in msecs) between checks by job proxies for job status on other cluster nodes.

- Import/Export job proxy timeout (msecs)
  - cmee.extframework.impexp.proxy.timeout
  - The value entered in the text field represents the time (in msecs) before a job proxy stops monitoring status of remote jobs on other clustered nodes.
This chapter describes the features of the Import Export tool and how to use it to import items into and export items from the Oracle API Catalog.

This chapter includes the following topics:

■ Overview
■ Using the Import/Export Tool
■ Starting the Import/Export Tool
■ Export Items from Oracle API Catalog
■ Importing Items into Oracle API Catalog

4.1 Overview

The Oracle API Catalog Import/Export Tool is designed to allow the interchange of assets and related metadata between instances of Oracle API Catalog.

---

**Note:** When using the Import/Export tool, adhere to the following best practices:

■ Exporting from Oracle API Catalog and importing to Oracle API Catalog is supported

■ Exporting from Oracle Enterprise Repository and importing to Oracle Enterprise Repository is supported

■ Exporting from Oracle API Catalog and importing to Oracle Enterprise Repository is supported

■ Exporting from Oracle Enterprise Repository and importing to Oracle API Catalog is supported only with use of the Export to API Catalog utility. You can use the Import/Export tool to import zip files created by this utility. Using the Import/Export tool in other ways to import assets from OER to OAC is NOT SUPPORTED and NOT RECOMMENDED.

See "Installing the Export to API Catalog Feature Patch" in *Oracle Fusion Middleware Installation Guide for Oracle Enterprise Repository* for information about installing the utility and "Exporting API Assets to API Catalog" in *Oracle Fusion Middleware Integration Guide for Oracle Enterprise Repository* for information about using the utility to export API assets from OER to OAC.

---
4.2 Using the Import/Export Tool

This section describes how you can use the Import/Export tool to import or export objects into or from Oracle API Catalog. This section contains the following topics:

- Section 4.2.1, “Overview”
- Section 4.2.2, “The parameter.properties File”

Note: The Import/Export tool requires Java Web Start. See Section 2.7, “Install Java Web Start” for more information about installing and configuring Java Web Start.

4.2.1 Overview

The Import/Export tool is available from within Oracle Enterprise Repository on the Admin page. The Oracle Enterprise Repository user is required to have the admin role to access the tool.

The Import/Export tool is a Java WebStart program that downloads and launches when invoked.

The Export operation enables you to search for objects in Oracle Enterprise Repository to export. You can search for and select single or multiple API assets to export. The result of the Export operation is a .zip file with the selected APIs. By default, a sample parameter.properties file is contained within the resulting .zip file.

Note: The Import/Export tool will not import or export the icons of APIs in Oracle API Catalog. Duplicate assets in the import bundle already present in OAC are ignored.

The Import operation takes a .zip file that was exported by the Import/Export tool and loads the contained items into Oracle API Catalog. The behavior for the Import is determined byMETA-INF/parameter.properties or a series of default actions.

Note: You can also import assets exported from the search results screen, as described in “Exporting Asset Search Results to ZIP” in Oracle Fusion Middleware Developer’s Guide for API Catalog.

Oracle API Catalog Administrators who must move data between different instances of Oracle API Catalog must use the Import/Export tool or the Oracle API Catalog OpenAPI. When the Import/Export tool is utilized, a parameter.properties file must be included in the .zip bundle before performing the import operation. The Export operation creates a placeholder file located in the /META-INF directory.

4.2.2 The parameter.properties File

Each import bundle contains a /META-INF directory. This directory contains a file that controls how the Import/Export tool behaves when the bundle is imported. If the parameter.properties file does not exist, it must be created before any Import operation.
4.2.2.1 Supported Parameters in the parameter.properties File

The supported parameters in the properties file are listed with the legal values for the parameter:

**PreserveUUIDs**
- **Used by:** Asset
- **Description:** This parameter retains the UUIDs of assets in the bundle rather than randomly generating new UUIDs
- **Values:**
  - `true` - UUIDs on newly created assets are UUID specified in the import bundle
  - `false` - UUIDs on newly created assets are randomly generated by Oracle API Catalog

**Standard.saver.assettype.props.duplicate.matching**

- **Used by:** Asset Type
- **Description:** This parameter specifies how the import tool identifies that an asset type in the bundle is a duplicate of something that already exists in Oracle API Catalog.
- **Values:**
  - **UUID** - A duplicate is determined if the UUID of the asset type specified in the import bundle matches the UUID of an asset type in Oracle API Catalog.
  - **Name** - A duplicate is determined if the Name of the asset type specified in the import bundle matches the Name of an asset type in Oracle API Catalog.

**Standard.saver.assettype.props.duplicate.handling**

- **Used by:** Asset Type
- **Description:** This parameter specifies how the import tool handles asset types in the bundle that are identified to be Duplicates of asset types in Oracle API Catalog.
- **Values:**
  - **Create New Version** - Create a new asset type with the UUID and archetype from the import bundle, but a name which is versioned with a suffix of a "V" and a number (for example, Service V2). The asset type has a new UUID.
  - **Ignore** - Ignore the duplicate asset type from the bundle. Do not import the asset type and do not modify the asset type in Oracle API Catalog.

Note: This parameter is not used in Oracle API Catalog, but it is present in the parameter.properties file. It is recommended to leave this property as the default value.
- **Merge** - Merge the asset type in Oracle API Catalog with the duplicate asset type from the bundle. The Merge operation does not modify any of the elements of the asset type in Oracle API Catalog. The operation is additive, appending elements in the asset type from the bundle to the asset type in Oracle API Catalog. Matching of the elements is by the XML Mapping of the element. The rules are:
  * If the element’s XML Mapping exists, then ignore the element.
  * If the element’s XML Mapping does not exist, then add it. The element is added by first locating the element’s tab, and then creating the tab if it does not exist. The element is then added to the end of the tab.

- **Overwrite** - Replace the asset type in Oracle API Catalog with the duplicate asset type from the bundle. The assets of the asset type in Oracle API Catalog is not modified, although the display of these assets is affected.

### Standard.saver.relationshiptype.props.duplicate.handling

**Note:** This parameter is not used in Oracle API Catalog, but it is present in the parameter.properties file. It is recommended to leave this property as the default value.

- **Used by:** Relationship Type
- **Description:** This parameter specifies how the import tool handles relationship types in the bundle that are identified to be Duplicates of relationship types in Oracle API Catalog. Duplicate relationship types are always matched by the Name and Direction characteristics of the relationship type.
- **Values:**
  - **Ignore** - Ignore the duplicate relationship type from the bundle. Create any asset relationships that use this relationship type and do not already exist in Oracle API Catalog.
  - **Overwrite** - Replace the relationship type in Oracle API Catalog with the duplicate relationship type from the bundle. This overwrites all information for the relationship type. This also creates any asset relationships that use this relationship type and do not already exist in Oracle API Catalog.

### Standard.saver.metadata.props.duplicate.handling

**Note:** This parameter is not used in Oracle API Catalog, but it is present in the parameter.properties file. It is recommended to leave this property as the default value.

- **Used by:** CMF Metadata
- **Description:** This parameter specifies how the import tool handles blocks of CMF Metadata in the bundle that are identified to be Duplicates of CMF Metadata in Oracle API Catalog. Duplicate CMF Metadata is always matched by the UUID of the metadata.
- **Values:**
  - **Ignore** - Ignore the duplicate CMF Metadata from the bundle.
- **Overwrite** - Replace the CMF Metadata in Oracle API Catalog with the duplicate CMF Metadata from the bundle. This overwrites all information for the CMF Metadata.

**Standard.saver.metadata.props.duplicate.handling**

**Note:** This parameter is not used in Oracle API Catalog, but it is present in the parameter.properties file. It is recommended to leave this property as the default value.

- **Used by:** CMF Metadata Entry Types
- **Description:** This parameter specifies how the import tool handles CMF Metadata Entry Types in the bundle that are identified to be Duplicates of CMF Metadata Entry Types in Oracle API Catalog. Duplicate CMF Metadata Entry Types are always matched by the Name of the metadata entry type.
- **Values:**
  - **Ignore** - Ignore the duplicate CMF Metadata Type from the bundle.
  - **Overwrite** - Replace the CMF Metadata Entry Type in Oracle API Catalog with the duplicate CMF Metadata Entry Type from the bundle. This overwrites all information for the CMF Metadata Entry Type.

**4.2.2.2 Sample parameter.properties File**

A sample parameter.properties file with all options included is as follows:

```properties
##Name: parameters.properties
##Location: import_bundle.zip/META_INF/

##Legend:
## a double ## indicates a comment
## a single # indicates a command which can be used in an import bundle
## the text between [ ]'s is the set of valid values for the parameter with comma
separating individual values

## for assets:
## This parameter will retain the UUIDs of assets in the bundle rather than
randomly generating new UUIDs
#PreserveUUIDs=[true,false]
#Standard.saver.asset.props.duplicate.handling=[Create New Version, Ignore, Overwrite]
#Standard.saver.asset.props.duplicate.matching=[UUID, NameVersion]

## for assettypes:
#Standard.saver.assettype.props.duplicate.handling=[Create New Version, Ignore, Merge, Overwrite]
#Standard.saver.assettype.props.duplicate.matching=[UUID, Name]

## for relationship types:
#Standard.saver.relationshiptype.props.duplicate.handling=[Ignore, Overwrite]

## For CMF metadata
#Standard.saver.metadata.props.duplicate.handling=[Ignore, Overwrite]

## For CMF metadata entry types
#Standard.saver.metadataentrytype.props.duplicate.handling=[Ignore, Overwrite]
```
4.3 Starting the Import/Export Tool

The Import/Export Tool can be launched from within Oracle API Catalog from the Admin screen. The Import/Export Tool requires JDK 1.4.2 or later.

4.3.1 Launching from Oracle API Catalog

The Import/Export Tool is launched from the Admin page. Due to the nature of the Import/Export Tool and the amount of data required for a comprehensive backup, the tool is restricted to users who are assigned the admin role, such as the built-in admin user.

The Import/Export section on the Admin page is enabled and configured by a property in system settings. If the section is not visible for a user with the admin role, then verify that the following system setting is enabled and set to True:

cmee.importexport.enabled

4.3.2 Launching from Outside of Oracle API Catalog

You may need to use the Import/Export tool when you aren’t logged in to Oracle API Catalog locally. You can save the Import Export tool and run it from your local machine.

To run the Import/Export utility from a local machine:

1. Follow the instructions in Section 2.7, "Install Java Web Start" to configure Java Web Start to run on the system
2. Click Import Export from the Admin page.
3. Click the Import/Export Client icon, select Save File or your browser’s equivalent, and then save the impexp.jnlp file.
4. Move the impexp.jnlp file to the location from which you want it to run.
5. Run the impexp.jnlp file.

**Note:** You can also run the Import Export tool by saving it locally, and then using the javaws executable, located in <JAVA_HOME>/jre/bin. For example:

```
<JAVA_HOME>/jre/bin/javaws -verbose /path_to_file/impexp.jnlp
```

4.3.3 Initial Startup

Upon launching the Import/Export Tool, you may be prompted to enter connection information for Oracle Enterprise Repository, as shown in Figure 4–1.

**Note:** You are not always prompted for credentials, except in the SSO environment.
The Oracle API Catalog Import/Export utility, showing the OER URL, the Username, and Password, with the Login and Cancel buttons.

- **OER URL**
  Path to the Oracle API Catalog installation, usually the same as the servlet path, as defined in the system settings. Example: `http://example.com/oer/`

- **Username**
  Name of a user who has been granted the Admin role within Oracle API Catalog.

- **Password**
  Password for the user specified above.

To set a maximum time in milliseconds that an Import/Export tool can run before it is automatically killed by the job monitor, a new system setting must be configured:

`cmee.extframework.impexp.monitor.maxruntime`

### 4.4 Export Items from Oracle API Catalog

This procedure is performed on the Oracle Enterprise Repository Admin screen.

1. Click **Import Export** from the Admin page.
2. Click the **Import/Export** icon. The Import/Export Client is displayed.
3. Click the **Export** tab.

**Figure 4–2  Oracle API Catalog Export Tab**

Select target and entity types to export...

Select Target File

Select Entities to Export

Assets

[Note: Select a file name to save the export to.
Select the Oracle API Catalog entities you would like to include within the export.]
The Export Tab, which asks you to select target and entity types to export (Assets is the only option for OAC). At the top is a list box to browse for target files, and in the middle a list of entities that you can select to export.

---

4. Enter an appropriate file name in the **Select Target File** field or click **Browse** to select the directory into which the target file will be exported. Click **Next**.

5. The **Select Assets to Export** dialog is displayed, as shown in Part 4–3, "Select Assets to Export Dialog".

**Figure 4–3  Select Assets to Export Dialog**

![Select Assets to Export Dialog](image)

The Export tab, prompting you to select assets to export. Towards the top of the page is **Find an asset**, where you can enter a keyword to search for, and **Available** and **Selected** columns for assets to be selected.

---

6. Use **Search** or click **List All** to display a list of items in the **Available** column.

7. Use the arrow buttons to move selected items between the **Available** and the **Selected** columns.

8. Click **Next**. A list of the files to be exported is displayed in the **Ready to Perform Export** dialog.

9. Click **Next**. A progress bar indicates the status of the export process.

---

**Note:** Variations in the size and complexity of the exported items affects the speed of this process and the behavior of the progress bar.

---

A summary of the export is displayed on completion, as shown in **Figure 4–4**.
4.5 Importing Items into Oracle API Catalog

This procedure is performed on the Oracle API Catalog Admin screen.

1. Click **Import Export** from the Admin page.
2. Click the **Import/Export** icon. The Import/Export Client is displayed.
3. Click the **Import** tab.
4. Enter the appropriate file name in the **Select File to Import** field or click **Browse** to select the source file for the import in Figure 4–5.

5. Click **Next**. The selected source file is listed in the Ready to Perform Import dialog is displayed, as shown in Figure 4–6.
6. Click **Next**. A progress bar indicates the status of the import process. Variations in the size and complexity of the imported items affects the speed of this process and the behavior of the progress bar.

   A summary of the import is displayed on completion, as shown in **Figure 4–7**.

**Figure 4–7 Performing Import Operation Dialog**

---

**Note:** When running the Import/Export tool, importing larger data sets may cause available memory issues. To avoid, save the `impexp.jnlp` file locally by right clicking on the Import/Export Client link and selecting “Save link as...” option from the context menu. Edit the `impexp.jnlp` file to increase the default max-heap-size amount to better accommodate the size of the data sets being imported. If the machine importing the data set has enough available memory, change the max-heap-size parameter to a higher value, as in the following example: `<j2se version="1.6" max-heap-size="1024m"/>`
This part describes advanced administration tasks for the lifecycle.
This part includes the following chapters:

- Chapter 5, "Configuring and Using Automated Harvesting"
- Chapter 6, "Editing Metadata of API Assets in Oracle API Catalog"
The tool used to populate Oracle API Catalog is called the Harvester. The Harvester reads metadata from Oracle products and deployed services. This includes Oracle SOA Suite and Oracle Service Bus.

The Harvester automatically creates assets and populates asset metadata based on the information in the harvested service.

The files are harvested as they are deployed to the runtime environment. The Harvester can be used from the command line, and within Ant and the Weblogic Scripting Tool (WLST).

This chapter contains the following sections:

- Section 5.1, "Getting Started with Harvester"
- Section 5.2, "Configuring the Harvester"
- Section 5.3, "Understanding Additional Harvester Information"

5.1 Getting Started with Harvester

This section describes how to get started with Harvester and its use in various high level use cases.

This section contains the following topics:

- Section 5.1.1, "Prerequisites"
- Section 5.1.2, "Harvester Functionality"
- Section 5.1.3, "Artifacts/Products Version Matrix"

5.1.1 Prerequisites

Before using the Harvester, you must perform the following prerequisites:

- If you wish to harvest from Oracle Service Bus, install the harvester to the same \$FMW_HOME to which Oracle Service Bus is installed or to a shared file system accessible to the Oracle Service Bus installation. See Section 5.2.2.2, "Selecting the Artifacts to Harvest for the Command Line" for more information.
Configuring the Harvester

**Note:** If you wish to harvest projects from Oracle Service Bus, you must use the osb-harvest.bat or osb-harvest.sh utilities instead of the usual harvest.bat or harvest.sh Harvester utilities. This requires Oracle Service Bus 12c libraries to be installed. See Chapter 5.2.5.3, "Harvesting Web Services from Oracle Service Bus" for more information about the OSB Harvester.

- When harvesting assets from deployed applications such as Oracle SOA Suite or Oracle Service Bus, then the application must be deployed on WebLogic 11g or higher.
- Harvester requires Java SDK version 6 or higher.

### 5.1.2 Harvester Functionality

You can use the Harvester to publish services exposed by the artifacts and products listed in **Table 5–1**.

### 5.1.3 Artifacts/Products Version Matrix

**Table 5–1** describes the version matrix of the artifacts/products that is supported with Harvester.

<table>
<thead>
<tr>
<th>Artifact/Product</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deployed WSDL</td>
<td>1.1</td>
</tr>
<tr>
<td>Deployed WADL</td>
<td>1.1.5</td>
</tr>
<tr>
<td>Oracle SOA Suite</td>
<td>11g and 12c (12.1.3 and 12.2.1)</td>
</tr>
<tr>
<td>Oracle Service Bus</td>
<td>11g and 12c (12.1.3 and 12.2.1)</td>
</tr>
<tr>
<td>WebLogic Web Services</td>
<td>11g and 12c (12.1.3 and 12.2.1)</td>
</tr>
</tbody>
</table>

**Note:** The harvester does not support harvesting end-to-end JSON or Native REST services in Oracle SOA Suite and Service Bus, introduced in the 12.2.1 release. Harvesting of other artifacts from the 12.2.1 releases of these products is supported.

### 5.2 Configuring the Harvester

This section describes how you can configure Harvester.

This section contains the following topics:

- **Section 5.2.1**, "Obtaining the Harvester"
- **Section 5.2.2**, "Configuring the Harvester for the Command Line"
- **Section 5.2.3**, "Invoking the Harvester Using the Repository.Submit Ant Task"
- **Section 5.2.4**, "Invoking Harvester from WLST"
- **Section 5.2.5**, "Runtime Harvesting Details"
- **Section 5.2.6**, "Performing Optional Harvester Configuration"
5.2.1 Obtaining the Harvester

The Harvesters are automatically unzipped into the following directories when installing Oracle API Catalog:

- `<FMW_HOME>/oer/tools/harvester`
- `<FMW_HOME>/oer/tools/osbharvester`

The Harvesters are also available in `12.1.3.0.0-OER-Harvester.zip`, which is bundled with the Oracle API Catalog 12c installation, at the following location:

- `<FMW_HOME>/oer/modules/tools/solutions/12.1.3.0.0-OER-Harvester.zip`

The OSB Harvester is also available in `12.1.3.0.0-OSB12c-Harvester.zip`, which is bundled with the Oracle API Catalog 12c installation, at the following location:

- `<FMW_HOME>/oer/modules/tools/solutions/12.1.3.0.0-OSB12c-Harvester.zip`

This manual refers to `<FMW_HOME>/oer/tools/harvester` or `<FMW_HOME>/oer/tools/osbharvester` as the `<Harvester Home>` directory, depending on which harvester you are using.

Table 5–2 describes the relationship matrix for the product and the corresponding version.

<table>
<thead>
<tr>
<th>Product and Version</th>
<th>Harvest</th>
<th>IDE Search, Browse, and Consume</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Bus 11g and 12c (12.1.3 and 12.2.1)</td>
<td>OER 12c</td>
<td>Yes</td>
</tr>
<tr>
<td>SOA Suite 11g and 12c (12.1.3 and 12.2.1)</td>
<td>OER 12c</td>
<td>Yes</td>
</tr>
<tr>
<td>WSDL</td>
<td>OER 12c</td>
<td>Varies</td>
</tr>
<tr>
<td>WADL</td>
<td>OER 12c</td>
<td>Varies</td>
</tr>
</tbody>
</table>

**Note:** The harvester does not support harvesting end-to-end JSON or Native REST services in Oracle SOA Suite and Service Bus, introduced in the 12.2.1 release. Harvesting of other artifacts from the 12.2.1 releases of these products is supported.

5.2.2 Configuring the Harvester for the Command Line

You can configure the Harvester from the command-line. Organizations can easily bootstrap their API Catalog installation using the Harvester from the command-line.

This section describes the tasks you need to perform to configure the Harvester for the command-line:

- **Section 5.2.2.1, "Setting Repository Connection Information for the Command Line"**
- **Section 5.2.2.2, "Selecting the Artifacts to Harvest for the Command Line"**
5.2.2.1 Setting Repository Connection Information for the Command Line

Open the XML file HarvesterSettings.xml located at <Harvester Home> and modify the following XML to point the Harvester to an Oracle API Catalog instance with the correct credentials:

```xml
<repository>
  <uri>http://localhost:7101/oer/</uri>
  <credentials>
    <user>smith</user>
    <password>*****</password>
    //To ensure security, the password must be encrypted.
    //The password encryption tool (encrypt.bat/encrypt.sh) allows you to encrypt
    //the passwords that are stored in the Harvester configuration
    //HarvesterSettings.xml file.
  </credentials>
  <timeout>30000</timeout>
</repository>
```

Note: It is recommended that you run the Harvester as a user with the admin role.

5.2.2.2 Selecting the Artifacts to Harvest for the Command Line

The Harvester can be run from the command line using the harvest.bat or harvest.sh utilities.

Note: You must use the OSB Harvester if you are harvesting projects from Oracle Service Bus. The OSB Harvester is installed into the <FMW_HOME>/oer/tools/osbharvester directory and can be run using the osb-harvest.bat or osb-harvest.sh utilities. The OSB Harvester must be installed into the same FMW_HOME to which OSB 12c is installed.

See Section 5.2.5.3, "Harvesting Web Services from Oracle Service Bus" for more information about using the OSB Harvester.

The harvest.bat and harvest.sh utilities can be run from the server to which Oracle API Catalog is installed. If you want to run the harvester from a different client, obtain the 12.1.3.0.0-OER-Harvester.zip file from the <FMW_HOME>/oer/modules/tools/solutions directory and extract the zip to any client that has a valid JAVA_HOME environment variable set, as discussed in Table 5–3.

The osb-harvest.bat and osb-harvest.sh utilities can be run from the server to which Oracle API Catalog is installed. If you want to run the harvester from a different client:

- Install Oracle Service Bus into a <FMW_HOME> directory on the client machine.
- Ensure that the JAVA_HOME environment variable is set on the client machine, as shown in Table 5–3.
- Obtain the 12.1.3.0.0-OSB12c-Harvester.zip file from the <FMW_HOME>/oer/modules/tools/solutions directory on the machine to which Oracle API Catalog is installed. Extract the file to the <FMW_HOME>/oer/tools directory on the target machine. The full path to the harvester after extracting the files should
be `<FMW_HOME>/oer/tools/osbharvester`. The OSB Harvester must be installed into the same `FMW_HOME` to which OSB 12c is installed.

Before running `harvest.bat`, `harvest.sh`, `osb-harvest.bat`, or `osb-harvest.sh`, ensure that the environment variables mentioned in Table 5–3 are set. In Windows, from a command window, you can type "set X" to see the value of the variable X, and "set X=abc" to set the value of X to "abc".

### Table 5–3 Command Line Script

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JAVA_HOME</td>
<td>Ensure that the JAVA_HOME environment variable points to an installed java runtime (JRE) or SDK. For Oracle Service Bus introspection, this must be Java version 6 or higher.</td>
</tr>
<tr>
<td>JAVA_OPTS</td>
<td>Optionally, set your JAVA_OPTS parameter to add any additional java parameters that are necessary. For example, if you need to use an HTTP proxy server, set the value to <code>-Dhttp.proxyHost=www-proxy.yourhost.com -Dhttp.nonProxyHosts= *.yourhost.com localhost</code> See Also: <a href="http://java.sun.com/javase/6/docs/technotes/guides/net/proxies.html">http://java.sun.com/javase/6/docs/technotes/guides/net/proxies.html</a></td>
</tr>
</tbody>
</table>

Table 5–4 shows the options that can be specified using the Harvester command line utility:

### Table 5–4 Command Line Options for the Harvester

<table>
<thead>
<tr>
<th>Harvester Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-settings&lt;file&gt;</code></td>
<td>Specifies the configuration settings XML file.</td>
</tr>
<tr>
<td><code>-url &lt;URL&gt;</code></td>
<td>Specifies the URL of Oracle API Catalog.</td>
</tr>
<tr>
<td><code>-user &lt;OER/OAC User Name&gt;</code></td>
<td>Specifies the user name of the Oracle API Catalog user.</td>
</tr>
<tr>
<td><code>-asset_version</code></td>
<td>Version to be applied to all assets created in the model</td>
</tr>
<tr>
<td><code>-file &lt;filename or URL&gt;</code></td>
<td>Specifies the file or directory to be harvested. This can be a filename or URL to the file.</td>
</tr>
<tr>
<td><code>-file_type &lt;type&gt;</code></td>
<td>Specifies the file type of the file to be harvested. If not specified, then the type is derived from the file extension. This must correspond to one of the filetypes in the config/plugins folder. By default, the following are supported: .bpel, .mfl, .policy, .wsdl, .xsd, .xquery, .xslt.</td>
</tr>
<tr>
<td><code>-remote_url &lt;URL&gt;</code></td>
<td>Specifies the running server from which to harvest the remote project, instead of from a file.</td>
</tr>
<tr>
<td><code>-remote_username &lt;username&gt;</code></td>
<td>Specifies the username to connect to the remote server.</td>
</tr>
<tr>
<td><code>-remote_server_type &lt;type&gt;</code></td>
<td>Specifies the type of remote server. Remote server could include either of the following: SOASuite, SOASuite11g, OSB, WLS. Note: Use SOASuite11g when running against a SOA Suite 11g server. Use SOASuite when running against a SOA Suite 12c server.</td>
</tr>
<tr>
<td><code>-remote_project &lt;type&gt;</code></td>
<td>Specifies the name of remote project to harvest, instead of a file. If omitted, all of the projects on the server are harvested. In the case of Oracle SOA Suite, this should be the name of the composite plus revision, for example, MyComposite_rev1. In the case of WLS, this should be the Application Name, as seen in the WebLogic Administration console and Enterprise Manager.</td>
</tr>
<tr>
<td><code>-soa_partition &lt;type&gt;</code></td>
<td>Specifies the name of partition for Oracle SOA Suite and by default, it uses the soa &quot;default&quot; partition.</td>
</tr>
<tr>
<td><code>-version</code></td>
<td>Specifies the print version information.</td>
</tr>
</tbody>
</table>
An example of a remote harvest is as follows:

```
harvest.bat -remote_url mysoasuiteserver:8001 -remote_username weblogic
-remote_server_type SOASuite -remote_project MyComposite_rev1.0 -soa_partition department1
```

Figure 5–1 shows the command line utility options and online help displayed by the harvest.bat -help command.

Command line options are required only when harvesting from an Oracle SOA Suite 11g server; the -remote_server_type SOASuite11g option must be included if this is the case. See Section 5.2.5.1, "Harvesting from Oracle SOA Suite Server" for more information.

If the options are omitted, then the Harvester uses the information in the HarvesterSettings.xml file in the <Harvester Home> directory, where harvest.bat resides. If options are specified on the command line, then these override the settings in HarvesterSettings.xml.

To ensure security, passwords are not passed via the command line and they must be encrypted. The only way to set passwords for Oracle API Catalog and remote servers is by storing them in the Harvester configuration (HarvesterSettings.xml) file. The password encryption tool (encrypt.bat/encrypt.sh) allows you to encrypt the passwords that are stored in the file. For more information about password encryption, see Chapter 8, "Password Encryption".
To point to the artifacts to be harvested using the HarvesterSettings.xml file in the <Harvester Home>, modify the following XML:

```
<query>
  <fileQuery>
    <rootDir>C:\samples</rootDir>
    <files>BPEL</files>
  </fileQuery>
</query>
```

or

```
<query>
  <fileQuery>
    <files>http://remote/server/my_generated_wsdl</files>
    <fileType>.wsdl</fileType>
  </fileQuery>
</query>
```

5.2.3 Invoking the Harvester Using the Repository.Submit Ant Task

You can invoke harvester as an Ant Task to ensure that all deployment information is stored in Oracle API Catalog at deployment time. You can use the repository.submit Ant task provided with the Harvester to harvest and import metadata into Oracle API Catalog. This task can be defined in the harvest-tasks.xml file, which is located in the <FMW_HOME>/oer/tools/harvester directory.

This section contains the following topics:

- Section 5.2.3.1, "Specifying Parameters for the repository.submit Ant Task"
- Section 5.2.3.2, "Specifying Parameters as Nested Elements"
- Section 5.2.3.3, "Running the Harvester from Ant"
- Section 5.2.3.4, "Using the Third-Party Tasks"

5.2.3.1 Specifying Parameters for the repository.submit Ant Task

Table 5–5 shows parameters that can be specified for the repository.submit Ant task in the harvest-tasks.xml file:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>repositoryURL</td>
<td>Repository instance to connect to.</td>
<td>Yes, unless specified by a property.</td>
</tr>
<tr>
<td>repositoryUsername</td>
<td>Username to log into Oracle API Catalog.</td>
<td>Yes, unless specified by a property.</td>
</tr>
<tr>
<td>repositoryPassword</td>
<td>Password to log into Oracle API Catalog. To ensure security, the password must be encrypted. The Oracle API Catalog Web console has a tool to encrypt passwords: <a href="http://%3Chost%3E:%3Cport%3E/%3Cdomain%3E/diag/encryptstrings.jsp">http://&lt;host&gt;:&lt;port&gt;/&lt;domain&gt;/diag/encryptstrings.jsp</a></td>
<td>Yes, unless specified by a property.</td>
</tr>
<tr>
<td>timeout</td>
<td>Number of seconds before calls to Oracle API Catalog will time out.</td>
<td>No. Defaults to 300 (5 minutes).</td>
</tr>
<tr>
<td>failOnError</td>
<td>Fails the entire build script if the Oracle API Catalog operation results in an error.</td>
<td>No. Defaults to &quot;true.&quot;</td>
</tr>
</tbody>
</table>
5.2.3.2 Specifying Parameters as Nested Elements

**FileSet**

FileSets are used to select sets of files to harvest. One or more fileSets must be specified.

The Harvester examines all the files selected by the fileSet, including files in .zip format (including .zip, .jar, and .ear files, for example).

In the Ant repository.submit task shown in Example 5–1, the Harvester examines all the files and directories under the fileSet directory (the /tmp/components directory) and imports them into Oracle API Catalog.

**URI**

URI identifies a file to introspect. One or more uris can be specified.
This can be an "http:" URL that points to a remote file, or a "file:" URL that points to a local file, or any other URL supported by java.

This can point to a single file, or a file in zip format that includes zips, jars, ears, etc. This could include exported Oracle Service Bus project jars. Oracle Service Bus projects can be exported from Oracle Service Bus Workshop, using the Export|Oracle Service Bus|Configuration Jar command.

The artifactStore attribute is the name of an Oracle API Catalog artifact store to look in. The artifact store must be created beforehand in the Oracle API Catalog Asset Editor. If specified, then the URI is resolved relative to the artifact store URL. When specifying a URI relative to an Artfact Store, the URI must resolve to a file such as a .wsdl or .zip file. The URIs that point to directories are not supported.

The fileType attribute indicates the file type of the file to be harvested. If not specified, then the type is derived from the file extension. This must correspond to one of the file types in the config/plugins folder. By default, the following are supported: .bpel, .mfl, .policy, .wsdl, .xsd, .xquery, .xslt.

**Example 5-1 Specifying the Files to Harvest with the Ant repository.submit Task**

```xml
<repository.submit repositoryurl="http://server.example.com:8080/oer"
            repositoryusername="myuser"
            registrypassword="******"
            settingsFile="../MyCustomSettings.xml">
  <files>
    <fileset dir="/tmp/components/">
      <include name="**/*"/>
    </fileset>
  </files>
</repository.submit>
```

**RemoteProject**

RemoteProject specifies that the Harvester should read a project from a remote server rather than from a file. The uri attribute indicates the running server from which to harvest the remote project. The username and password attributes indicate login information for the remote server.

The serverType attribute indicates the remote server type, which could be either Oracle SOA Suite (SOASuite or SOASuit11g), Oracle Service Bus (OSB) or WebLogic Server (WLS). The projectName element indicates the name of remote project to harvest, instead of a file. If omitted, all of the projects on the server are harvested. In the case of Oracle SOA Suite, this should be the name of the composite plus revision, for example, MyComposite_rev1. In the case of WLS, this should be the Application Name, as seen in the WebLogic Administration console and Enterprise Manager.

**Example 5-2 Specifying Parameters as Nested Elements - RemoteProject**

```xml
<repository.submit repositoryurl="http://server.example.com:8080/oac"
            repositoryusername="myuser"
            registrypassword="******"
            settingsFile="../MyCustomSettings.xml">
  <remoteProjects url="http://mywlsserver:7001" username="admin" password="*****"
            serverType="SOASuite" soaPartition="${partition}">
```
// To ensure security, the password must be encrypted.
// The password encryption tool (encrypt.bat/encrypt.sh) allows you to encrypt
// the passwords that are stored in the Harvester configuration
// (HarvesterSettings.xml) file.
<projectName>MySOAComposite1_rev1.0</projectName>
<projectName>MySOAComposite3_rev2.0</projectName>
</remoteProjects>
</repository.submit>

Producing Projects
You can specify multiple projects tagged as producing projects for all assets that are
created from the harvest. The projects must already exist in OER.

Within the <repository.submit> element, include:

■ <producingProject name="MyProjectName"/>
■ <producingProject name="MyOtherProjectName"/>

Producing projects may also be set within the HarvesterSettings.xml file, in the same
way.

5.2.3.3 Running the Harvester from Ant
To import the Harvester Ant tasks, include a line, as follows, in your Ant XML:
<taskdef file=${harvester.dir}/harvest-tasks.xml/>

where harvester.dir is the <FMW_HOME>/oer/tools/harvester
directory.

When running from the command line, ensure that the Harvester libraries are
available to Ant’s classpath. Harvester includes a script called runant that sets up the
ant classpath correctly. This script must be used to launch ant when using the
harvester ant tasks. For example:
runant -f mybuild.xml

(where mybuild.xml is your ant build XML).

Any arguments passed to runant are passed along to ant. Before running harvest.bat
or harvest.sh, ensure that the required environment variables are set. Refer to
Table 5–6 for the list of environment variables that you need to set.

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
</tr>
</thead>
</table>
| JAVA_HOME            | Ensure that the JAVA_HOME environment variable points to an installed java runtime
                     | (JRE) or SDK. For Oracle Service Bus introspection, this must be Java version 6 or
                     | higher. |
Configuring the Harvester

5.2.3.4 Using the Third-Party Tasks

The runant script uses ant's old launcher: org.apache.tools.ant.Main. The new launcher uses a URIClassLoader interface, which interferes with the usage of custom URISreamHandlers of the Oracle Service Bus.

The old ant launcher does not support automatic discovery of custom ant tasks in the ant/lib directory. When defining custom tasks with <taskdef>, you must specify a classpath attribute.

For example, the following is an example of how the <taskdef> should NOT be defined:

```xml
<taskdef id="ant-contrib"
resource="net/sf/antcontrib/antcontrib.properties"/>
```

The following is an example of how the <taskdef> should be defined:

```xml
<taskdef id="ant-contrib"
resource="net/sf/antcontrib/antcontrib.properties"
classpath="${ant.home}\lib\ant-contrib.jar"/>
```

5.2.4 Invoking Harvester from WLST

WLST (WebLogic Scripting Tool) is a command-line scripting interface that system administrators can use to manage WebLogic Server instances. WLST supports Oracle Service Bus 11g and 12c and SOA Suite 11g and 12c.

For more information about WLST, see Oracle Fusion Middleware WebLogic Scripting Tool Command Reference.

You can invoke harvester from WLST to ensure that all deployment information is stored in Oracle API Catalog at deployment time. Perform the following steps to invoke harvester from WLST:

1. Copy the oer.py file from the harvester installation folder to the <BEA_HOME>\wlserver\common\wlst\lib directory.

2. In the command window, enter the following commands:
   - Initialize the WLST classpath: run <BEA_HOME>\wlserver\server\bin\setWLSEnv.cmd
   - Initialize the Harvester classpath: cd to the <harvester> folder and run <harvester>\setenv.bat

---

### Table 5–6 (Cont.) Command Line Script Environment Variable Description

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANT_HOME</td>
<td>Ensure that the ANT_HOME environment variable points to an installation of Apache Ant, version 1.6.2 or higher. For Oracle Service Bus introspection, ensure that the Ant version is 1.6.5 or higher.</td>
</tr>
<tr>
<td>BEA_HOME</td>
<td>Ensure that the BEA_HOME environment variable points to the installation directory containing Oracle Service Bus server, if you plan to harvest projects from Oracle Service Bus. For example, C:\bea.</td>
</tr>
<tr>
<td>JAVA_OPTS</td>
<td>Optionally, set your JAVA_OPTS parameter to add any additional java parameters that are necessary. For example, if you need to use an HTTP proxy server, set the value to -Dhttp.proxyHost=www-proxy.yourhost.com -Dhttp.nonProxyHosts= **\yourhost.com</td>
</tr>
</tbody>
</table>

---

Ensure that the ANT_HOME environment variable points to an installation of Apache Ant, version 1.6.2 or higher. For Oracle Service Bus introspection, ensure that the Ant version is 1.6.5 or higher.

Ensure that the BEA_HOME environment variable points to the installation directory containing Oracle Service Bus server, if you plan to harvest projects from Oracle Service Bus. For example, C:\bea.

Optionally, set your JAVA_OPTS parameter to add any additional java parameters that are necessary. For example, if you need to use an HTTP proxy server, set the value to -Dhttp.proxyHost=www-proxy.yourhost.com -Dhttp.nonProxyHosts= **\yourhost.com |localhost* See Also: http://java.sun.com/javase/6/docs/technotes/guides/net/proxies.html
3. To invoke WLST from ant, perform the following steps:

- In the command prompt, change to the `<harvester>` directory by using the `cd` command.
- Enter `runant.bat -f<antscriptname>`, where the ant script invokes the `weblogic.WLST.java` class using ant's `<java>` task.
- The `<java>` task should set `fork = true`.
- The `<java>` task should include the environment's `CLASSPATH` (`runant.bat` will set `CLASSPATH` to include the harvester classes)
- The `<java>` task should also include on the classpath any libraries referenced by the WLST script.

The sample ant script is as follows:

```xml
<property environment="env"/>
<target name="harvest">
  <java classname="weblogic.WLST" fork="true">
    <arg line="${domain.import.script}"/>
    <arg line="${import.config.file}"/>
    <classpath refid="extra.class.path"/> <!--extra jars used by WLST script, e.g. OSB jars-->
    <classpath path="${env.CLASSPATH}"/>
  </java>
</target>
```

The `oer.harvest()` method takes one argument: a dictionary of command line parameters. These are the same parameters that are available from the harvester command line.

For more information about using the harvester from command line, see Section 5.2.2.2, "Selecting the Artifacts to Harvest for the Command Line".

A sample usage is as follows:

```python
argMap = {}
argMap['-harvester_home'] = '.'
argMap['-bea_home'] = 'c:/bea'
argMap['-remote_url'] = 'http://mywlsserver:7001'
argMap['-remote_username'] = 'admin'
argMap['-remote_password'] = '*****'
//To ensure security, the password must be encrypted.
//The password encryption tool (encrypt.bat/encrypt.sh) allows you to encrypt
//the passwords that are stored in the Harvester configuration
argMap['-remote_server_type'] = 'SOA Suite'
argMap['-remote_project'] = 'MySOAComposite_rev1.0'
argMap['-soa_partition'] = 'department1'
oer.harvest(argMap)
```

The `-harvester_home` argument must point to the location of the harvester installation. The `-bea_home` argument is required for Oracle Service Bus harvesting.

### 5.2.5 Runtime Harvesting Details

As assets move from development to testing, staging, and production environments, you may want to harvest these assets into Oracle API Catalog, so that Oracle API
Catalog has the most up-to-date endpoints. Oracle API Catalog overwrites the existing endpoints of existing assets in Oracle API Catalog. If the WSDL is modified as the asset moves through its lifecycle, the updated WSDL is also associated with the existing asset in Oracle API Catalog.

This section describes the runtime harvesting details for different servers. This section contains the following topics:

- Section 5.2.5.1, "Harvesting from Oracle SOA Suite Server"
- Section 5.2.5.2, "Harvesting Web Services from WebLogic Server"
- Section 5.2.5.3, "Harvesting Web Services from Oracle Service Bus"
- Section 5.2.5.4, "Harvesting Deployed Services"

5.2.5.1 Harvesting from Oracle SOA Suite Server

When an Oracle SOA Suite project deployed to a running server is introspected, any services exposed by the composite results in the creation of API Assets in Oracle API Catalog.

You need the following WLS security roles when harvesting from Oracle SOA Suite server:

- Admin
- Operator
- Monitor

The `-remote_url` parameter should point to the port of the soa-infra managed server. The default value of this in Oracle SOA Suite is 8001.

The soa-infra managed server must be up and running. Harvesting connects to the MDS database as part of the remote harvesting. The MDS database must be running and accessible from the machine where harvesting is taking place.

The SSL/HTTP protocol is supported. Harvester connects to the server MBeans via the t3 protocol.

The harvester automatically saves the value of `-remote_url` in the Deployment URI property, in the assets harvested from the runtime servers.

An example for harvesting from Oracle SOA Suite server is as follows:

```
harvest.bat -remote_url mysoasuiteserver:8001 -remote_username weblogic
-remote_server_type SOASuite -remote_project MyComposite_rev1.0 -soa_partition default
```

An example for harvesting from Oracle SOA Suite 11g server is as follows:

```
harvest.bat -remote_url mysoasuiteserver:8001 -remote_username weblogic
-remote_server_type SOASuite11g -remote_project MyComposite_rev1.0 -soa_partition default
```

To ensure security, passwords are not passed via the command line and they must be encrypted. The only way to set passwords for Oracle API Catalog and remote servers is by storing them in the Harvester configuration (`HarvesterSettings.xml`) file. The password encryption tool (`encrypt.bat/encrypt.sh`) allows you to encrypt the passwords that are stored in the file. For more information about password encryption, see Chapter 8, "Password Encryption".

The Oracle SOA Suite server has also implemented partitions to further categorize composites during runtime. This enables you to select the partition to use, along with
the composite name, in the harvester. The parameter name in the HarvesterSettings.xml file is called soaPartition. A sample HarvesterSettings.xml file is as shown below:

```xml
<remoteQuery>
  <serverType>SOASuite</serverType>
  <projectName>MyComposite_rev1.0</projectName>
  <uri>http://remotehost:8001/</uri>
  <credentials>
    <user>weblogic</user>
    <password>password</password>
  </credentials>
  <soaPartition>partition_name</soaPartition>
</remoteQuery>
```

**Note:** For Oracle SOA Suite 11g, you must set the `<serverType>` element to SOASuite11g. Additionally, you must include the `-remote_server_type SOASuite11g` command line argument when harvesting from Oracle SOA Suite 11g server.

**Note:** Partitions should be used only with SOA servers of version 11g R1 PS2 or later. You should comment the `soaPartition` element, if you are using a version earlier to PS2.

By default, the Harvester harvests using the `partition_name` option set to "default". If it is not specified, then partition "default" comes out-of-the-box by SOA deployments.

**Note:** You can deploy the same composite under different partitions in SOA Suite.

If this occurs and you harvest more than one of them, only one API asset gets created in OAC and will continually be overwritten by the latest harvest.

---

**Harvesting SOA Suite Composites in a Cluster**

The OHS server is deployed as a part of the WebTier label into the Middleware Home and is associated with the domains configured within this Middleware Home, as well.

There is a number of URI’s that would normally be assigned to the OHS server’s `mod_wl_2x.so` HTTP server module, some of which can be seen below (taken from the OHS `mod_wl_ohs.conf` file and modified in order to make it generic for external consumption):

```
LoadModule weblogic_module
"/opt/Oracle/Middleware/wls10.3/server/plugin/linux/x86_64/mod_wl_22.so"
<IfModule weblogic_module>
  ConnectTimeoutSecs 10
  ConnectRetrySecs 2
  Debug ALL
  WLLLogFile /tmp/weblogic.log
  DebugConfigInfo ON
  WLSocketTimeoutSecs 2
  WLIOTimeoutSecs 300
</IfModule>
```
Idempotent ON
FileCaching ON
KeepAliveSecs 20
KeepAliveEnabled ON
DynamicServerList ON
WLProxySSL OFF
</IfModule>

<Location /soa-infra>
SetHandler weblogic-handler
WebLogicCluster slc01ese.example.com:23049,wls02.example.com:15980
</Location>
<Location ~ "/bea_wls_internal/iiop/Client***">
SetHandler weblogic-handler
WebLogicCluster wls01.example.com:23049,wls02.example.com:15980
</Location>
<Location ~ "/bea_wls_internal/HTTPClnt***">
SetHandler weblogic-handler
WebLogicCluster wls01.example.com:23049,wls02.example.com:15980
</Location>
<Location /console>
SetHandler weblogic-handler
WebLogicHost wls01.example.com
WebLogicPort 19617
</Location>
<Location /em>
SetHandler weblogic-handler
WebLogicHost wls01.example.com
WebLogicPort 19617
</Location>
<Location /bea_wls_internal>
SetHandler weblogic-handler
WebLogicHost wls01.example.com
WebLogicPort 19617
</Location>
<Location /jndi>
SetHandler weblogic-handler
WebLogicHost wls01.example.com
WebLogicPort 23049
</Location>
<Location /HTTPClnt>
SetHandler weblogic-handler
WebLogicHost wls01.example.com
WebLogicPort 23049
</Location>

These special URI patterns allow for JMS and JMX requests to be routed through the OHS WebLogic module.

Other than any changes that you had applied to code, these are the only customizations that had been applied to the SOA clustered installation.

### 5.2.5.2 Harvesting Web Services from WebLogic Server

An Oracle Service Bus project can be submitted to Oracle API Catalog from the command line. The project must be exported from Oracle Service Bus Workshop, using the command `Export|Oracle Service Bus: Configuration jar`. 
The harvester will handle Proxy Services and WSDL artifacts. The resulting asset will be an API asset type.

You do not need any WebLogic Server security roles when harvesting from WebLogic Server.

The `-remote_url` parameter should point to the port of the WLS admin server. The default value of this in Weblogic is 7001. The WLS admin server must be up and running. The SSL/HTTP protocol is supported. Harvester connects to the server MBeans via the t3 protocol.

The harvester automatically saves the value of `-remote_url` in the Deployment URI property, in the assets harvested from the runtime servers.

### 5.2.5.3 Harvesting Web Services from Oracle Service Bus

You need the Admin WLS security role when harvesting Web Services from Oracle Service Bus.

You must use the OSB Harvester to harvest web services from Oracle Service Bus. The OSB Harvester is installed in the `<FMW_HOME>/oer/tools/osbharvester` directory.

The OSB Harvester can be run from the command line using the `osb-harvest.bat` or `osb-harvest.sh` utilities. See Section 5.2.2, "Configuring the Harvester for the Command Line" for information about configuring the OSB Harvester.

**Note:** When harvesting from a Service Bus 11g server using the `osb-harvest.bat/.sh` utility, the Service Bus harvester must be run from the same Middleware home into which Service Bus 12c is installed. Even though the Service Bus server is 11g, the Harvester requires the Service Bus 12c libraries. There are multiple ways to accomplish this; the easiest way is to copy the Service Bus harvester to an existing Service Bus 12c server.

If Service Bus 12c is installed on a separate client or in a separate Middleware home, you can update the `setenv.sh` or `setenv.bat` script to account for this location.

The `-remote_url` parameter should point to the port of the WLS admin server for the Oracle Service Bus domain. The default value of this in Weblogic is 7001. The WLS admin server must be up and running. The SSL/HTTP protocol is supported. Harvester connects to the server MBeans via the t3 protocol.

The harvester automatically saves the value of `-remote_url` in the Deployment URI property, in the assets harvested from the runtime servers.

An example of harvesting Web Services from Oracle Service Bus is as follows:

```
osb-harvest.bat -remote_url myosbsrver:7001 -remote_username weblogic -remote_server_type OSB -remote_project MyOSBProject
```

To ensure security, passwords are not passed via the command line and they must be encrypted. The only way to set passwords for Oracle API Catalog and remote servers is by storing them in the Harvester configuration (`HarvesterSettings.xml`) file. The password encryption tool (`encrypt.bat/encrypt.sh`) allows you to encrypt the passwords that are stored in the file. For more information about password encryption, see Chapter 8, "Password Encryption".
5.2.5.4 Harvesting Deployed Services

In addition to harvesting APIs from SOA Suite and Service Bus, you can also harvest deployed services using a URL to a deployed WSDL file. The example below will harvest a Weather service from the internet:


This method can be used to harvest secure endpoints from Oracle API Gateway (OAG). To get the WSDL URL from OAG, make sure the OAG Policy Studio publishes its WSDL to clients by checking "Allow the Gateway to publish WSDL to clients" checkbox on the WSDL tab in the Service Handler for the web service.

5.2.6 Performing Optional Harvester Configuration

You can optionally modify these additional configuration settings in the XML file `HarvesterSettings.xml` in the `<Harvester Home>` directory:

- `<harvesterDescription>`: A description about the harvesting performed. This information is stored in the Harvester Properties of the assets created in Oracle API Catalog.
- `<harvesterVersion>`: A version of the harvesting performed. This information is stored in the Harvester Properties of the assets created in Oracle API Catalog.
- `<namespace>`: A namespace that is added to abstract (non-artifact) Oracle API Catalog assets that are created during harvesting. The namespace is used in duplicate detection. If left empty, then this is set based on information from Oracle SOA Suite and Oracle Service Bus projects when available. That is, generally, the best practice, so override this with caution.
- `<workDir>`: A temporary directory where the zip and jar files are unzipped. By default, the system temp directory is used.

5.2.7 Configuring Logging for the Harvester

The Harvester uses log4j for logging the detailed tasks performed and the log file is placed in the `<Harvester Home>` directory. The logging options can be changed by updating the `log4fl.properties` file located in the `<Harvester Home>` directory.

5.2.8 Transaction Handling in Harvester

By default, the harvester makes all of its changes to Oracle API Catalog in a single transaction. Transactional operations in the harvester function is based on the following rules:

- An Oracle API Catalog server only supports one transaction at a time. If a transaction cannot be started, then the harvester informs you about it.
- There is a set of timeouts associated with the transactions. These will terminate a harvester operation if the timeouts are exceeded.
  - The setting, `cmeex.extframework.impexp.monitor.rex.maxidle`, specifies the maximum amount of time the transaction is kept alive if it loses connectivity with the client. This allows the transaction to be cancelled if the client-side harvester process is killed.
  - The setting, `cmeex.extframework.impexp.monitor.maxruntime`, specifies the maximum amount of time that the entire transaction will take.
It is possible that a single operation will exceed the cmee.extframework.impexp.monitor.rex.maxidle (maxidle) setting and cause all subsequent operations to fail. When this occurs, the last error in the log will state that ‘An error occurred while attempting to rollback because a transaction has not been started.’. This is because the transaction automatically rolls back when the statement exceeds the maxidle and the transaction is no longer available when the client application attempts a forceful rollback, having detected errors. The result is that no data, from the harvester operation, is added to Oracle API Catalog.

5.3 Understanding Additional Harvester Information

This section describes additional information about using the Harvester.

This section contains the following topics:

- Section 5.3.1, “Previewing of the Created Assets in Harvester”
- Section 5.3.2, “Best Practices”
- Section 5.3.3, “Known Issues”

5.3.1 Previewing of the Created Assets in Harvester

You can use the Preview feature to view the created assets in harvester. You can use the Preview feature using either of the following methods:

Using Command Line

At the command line or in an ant task, if you add “-preview true”, then the Harvester runs and displays a list of all of the assets that would be created, but it does not actually commit the changes.

Using the HarvesterSettings.xml File

You can also set the preview mode in the HarvesterSettings.xml file as follows:

```xml
<introspection preview="true">
  <reader>com.oracle.oer.sync.plugin.reader.file.FileReader</reader>
  <writer>com.oracle.oer.sync.plugin.writer.oer.OERWriter</writer>
</introspection>
```

5.3.2 Best Practices

This section describes best practices for the Harvester. It contains the following topics:

- Section 5.3.2.1, "Recommended Privileges for Harvesting"
- Section 5.3.2.2, "Do Not Override the Namespace Parameter"
- Section 5.3.2.3, "Namespaces in WSDL Files"

5.3.2.1 Recommended Privileges for Harvesting

Only users with the admin role can harvest assets.

5.3.2.2 Do Not Override the Namespace Parameter

Harvester offers you with the ability to set the project namespace for the assets being harvested. The project namespace is used in detecting duplicate assets, as different namespaces will result in different assets. In most cases, you are recommended not to override this parameter.
However, harvester automatically sets the project namespace based on the Oracle SOA Suite or Oracle Service Bus project name when harvesting from those products, which is the recommended behavior.

This is especially important to follow, when you harvest the same project and revision from multiple sources (JDev and runtime). Changing or overriding the namespace leads to duplicate assets in OER.

5.3.2.3 Namespaces in WSDL Files
In the WSDL files that you harvest, it is recommended that you use a unique namespace for each unique interface, service, and endpoint.

Correlation to existing assets in the Oracle API Catalog is done through QNames, so if you make significant changes to interface, service, or endpoint assets and do not change the QNames, you will overwrite the existing asset with the modified asset.

Table 5–7 shows the correlation of Oracle API Catalog assets with WSDL structure:

<table>
<thead>
<tr>
<th>Repository Asset</th>
<th>WSDL Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service</td>
<td>/definition/service/@name</td>
</tr>
<tr>
<td>Endpoint</td>
<td>/definition/service/port/@name</td>
</tr>
<tr>
<td>Interface</td>
<td>/definition/portType/@name</td>
</tr>
</tbody>
</table>

5.3.3 Known Issues
This section describes the following known Harvester issues:

- **Section 5.3.3.1, "Supported Remote Server Types"**
- **Section 5.3.3.2, "Versioning of Assets"**
- **Section 5.3.3.3, "Using Incorrect Encrypted Password"**
- **Section 5.3.3.4, "View in Repository Option"**

5.3.3.1 Supported Remote Server Types
Currently, the Harvester only supports connecting to the remote Oracle SOA Suite, Oracle Service Bus, and WebLogic Server projects that are running on WebLogic 11g and 12c.

5.3.3.2 Versioning of Assets
The Harvester derives the asset names from the artifact it introspects (WSDL, WADL, Composite). If you harvest an artifact that already exists in OAC, the harvester will simply overwrite the existing one.

However, if the artifact has changed in any way but the name convention is the same as something that already exists, such as a service name + namespace from a WSDL, the harvester creates a new asset and appends a number to the end of the asset’s name, such as "-2", "-3", "-4" and so on.

You can manually remove the old asset from OAC and rename the new one.

5.3.3.3 Using Incorrect Encrypted Password
If you configure the HarvesterSettings.xml file with incorrect encrypted password, then a long stack trace is displayed, which is as follows:
log4j:WARN No appenders could be found for logger (org.apache.axis.i18n.ProjectResourceBundle).
log4j:WARN Please initialize the log4j system properly.
0 com.oracle.oer.sync.framework.MetadataIntrospectionException:
0 com.oracle.oer.sync.framework.MetadataIntrospectionException: Unable to read plugin file:
C:\\Drive-E\XU-harvest-tools\dec16\OER-Harvester\.\plugins\biz.introspector
at 0 com.oracle.oer.sync.framework.MetadataManager.init(MetadataManager.java:308)
  at com.oracle.oer.sync.framework.Introspector.<init>(Introspector.java:188)
  at com.oracle.oer.sync.framework.MetadataManager.init(MetadataManager.java:306)
... 2 more
Caused by: com.oracle.oer.sync.framework.MetadataIntrospectionException: Unable to read plugin file:
C:\\Drive-E\XU-harvest-tools\dec16\OER-Harvester\.\plugins\biz.introspector
at 0 com.oracle.oer.sync.framework.impl.DefaultPluginManager.processIntrospector(DefaultPluginManager.java:127)
at 0 com.oracle.oer.sync.framework.impl.DefaultPluginManager.<init>(DefaultPluginManager.java:73)
at 0 com.oracle.oer.sync.framework.MetadataManager.init(MetadataManager.java:306)
... 2 more
Caused by: java.lang.IllegalArgumentException: The char '0x12' in 'java.lang.IllegalArgumentException: The char '0x12' in 'E/?↕?↕H?rd' is not a valid XML character.' is not a valid XML character.
at 0 org.apache.axis.components.encoding.AbstractXMLEncoder.encode(AbstractXMLEncoder.java:110)
at org.apache.axis.utils.XMLUtils.xmlEncodeString(XMLUtils.java:131)
at org.apache.axis.AxisFault.printStackTrace(AxisFault.java:796)
at com.oer.log4j.spi.ThrowableInformation.getThrowableStrRep(ThrowableInformation.java:42)
at com.oer.log4j.spi.ThrowableInformation.getThrowableStrRep(ThrowableInformation.java:217)
at com.oer.log4j.WriterAppender.subAppend(WriterAppender.java:298)
at com.oer.log4j.RollingFileAppender.subAppend(RollingFileAppender.java:294)
at com.oer.log4j.WriterAppender.append(WriterAppender.java:157)
at com.oer.log4j.AppendersSkeleton.doAppend(AppendersSkeleton.java:251)
at com.oer.log4j.helpers.AppenderAttachableImpl.appendLoopOnAppenders(AppenderAttachableImpl.java:57)
at com.oer.log4j.Category.callAppenders(Category.java:255)
at com.oer.log4j.Category.forcedLog(Category.java:445)
at com.oer.log4j.Category.log(Category.java:682)
at 0 com.oracle.oer.sync.framework.logger.DefaultMetadataLogger.error(DefaultMetadataLogger.java:98)
at 0 com.oracle.oer.sync.plugin.writer.oer.OERConnectionCache.getAuthToken(OERCon
5.3.3.4 View in Repository Option

From Oracle JDeveloper 12c, when you click the View in Repository option, it opens up the asset details in JDeveloper, which is for read-only purpose and the links within this page will not work.

As a workaround, you may need to login to Oracle API Catalog console and click links for further information.
This chapter describes editing the metadata of harvested API assets in Oracle API Catalog and publishing API assets so users with the developer role can discover and use them.

After an API asset is harvested, as described in Chapter 5, "Configuring and Using Automated Harvesting", an Oracle API Catalog user with the curator or admin role can edit its metadata to facilitate a developer’s discovery and understanding of an API. For instance, a curator can add keywords that make the asset more visible when searching for specific key words. The curator can also add a detailed description and a link to documentation for each API harvested and added to Oracle API Catalog.

For more information about the roles discussed in this section, see the "Role-Based Features" section in the Oracle Fusion Middleware Concepts Guide for Oracle API Catalog.

This chapter contains the following sections:

- Editing the Metadata of an API Asset
- Publishing an API Asset
- Using EasyLinks
- Deleting an Asset

### 6.1 Editing the Metadata of an API Asset

A user with the curator role should add metadata to API assets after they are harvested.

To edit the metadata of an API asset:

1. Search for an API asset for which you want to edit the metadata. See the "Finding Assets" section in the Oracle Fusion Middleware Developer’s Guide for API Catalog for more information about searching for APIs.

2. Click an API on the Search Results page to open its detail page.

3. Click the Edit icon, as shown in Figure 6–1. The API’s Edit Asset page is displayed.
4. Add any new metadata (or edit existing metadata) associated with the API. The metadata you can edit are described as follows:

- **Name:** The name of the API asset as it appears in Oracle API Catalog. The harvester automatically populates this field. You do not have to keep the default name; you can change this to anything that makes sense for this particular API. This field is required.

- **Version:** The version of the API asset. The harvester supplies a default version based on information that it finds in the composite; if no version information is found in the composite, the harvester assigns it a default version. You can change this to anything that makes sense for this particular API.

- **Keywords:** Keywords that users can use to find this asset in search results. You can also include name value pairs, such as platform=linux. Separate each keyword with a comma.

  **Note:** The maximum number of characters that can be entered into the Keyword field is 4000.

- **API Status:** The published status of the API. When an API is initially harvested, this is set to Draft by default. When you want to publish this asset to make it discoverable by developers, select Published from this list. See Section 6.2, "Publishing an API Asset" for more information about publishing an API asset.

- **Active Status:** The Active Status of an API. Active is selected by default. Select Retired when an API is retired.

  **Note:** When an API is retired, it will not be discoverable by developers, regardless of the published status.

- **Description:** The detailed description of the API's functionality, including functionality and other relevant information about using the API, such as a secure endpoint URL.
You can use HTML tags to add links or new elements, such as a contacts element, in the description field. For example, adding `<b>Contacts:</b> John Smith` to the description field adds a searchable Contacts element containing the name John Smith.

The cmee.asseteditor.enable-embedded-html-asset-description system setting must be enabled for HTML tags to function in this field.

---

**Note:** The maximum number of characters that can be entered into the Description field is 4000.

---

- **Documentation URL:** The URL to documentation for this API. You can link to a single document, a wiki page, or another page that collects multiple doc links. This URL will become a clickable link on the API's asset detail page.

- **Icon:** Select one of the following:
  - **Icon URL:** The URL pointing to the icon that you want to use for this API. Click **Preview** to display a preview of the icon.
  - **Upload Icon:** Click **Browse** to browse for an icon on your local machine to use for this API.

5. Click **Save** to save your changes.

### 6.2 Publishing an API Asset

After being harvested, an API asset is not discoverable by developers. A user with the curator or admin role must publish the API to make it visible to developers.

To publish an API:

1. Search for an API asset that you want to publish. See the "Finding Assets" section in the *Oracle Fusion Middleware Developer's Guide for API Catalog* for more information about searching for APIs.

   **Tip:** Select **Draft** from the list to return only APIs that have not yet been published.

2. Click an API on the **Search Results** page to open its detail page.
3. Click the **Edit** icon. The API's Edit Asset page is displayed.
4. From the **API Status** list, select **Published**.
5. Click **Save** to save your changes. Users with the developer role can now discover this asset by search. This API will also be displayed in the **Recently Published APIs** section of the Oracle API Catalog home page.

### 6.3 Using EasyLinks

EasyLinks may be useful if you want to add a link to an API detail page in documentation or on another API detail page.

The *EasyLink* icon is displayed on API pages, as shown in Figure 6–2. These icons open a pop-up window that displays a URL to the API detail page that can copied to the clipboard and pasted into documents or email messages as pointers to specific APIs.
To add a link to an API detail page:
1. Navigate to the detail page of the API that you want to create a link to.
2. Click the EasyLink icon.
3. Paste the URL from the pop-up window to where you want to create the link.
4. If adding the link to the Description field of an API’s detail page, edit the link so that it opens in a new tab by adding a target="_blank" attribute to the link.

6.4 Deleting an Asset

To delete an asset in Oracle API Catalog, perform the following steps:

---

**Note:** Only users with the admin role can delete assets from Oracle API Catalog.

---

1. Perform a search for the API asset(s) that you want to remove from Oracle API Catalog.
2. To delete an API:
   - To delete an API from the Search Results screen, select the check box next to each API that you want to delete, and then click **Delete**. You can delete multiple APIs simultaneously using this method.
   - To delete an API from its detail page, open the detail page for the API you want to delete, and then click **Delete**.
Part IV

Securing Oracle API Catalog

This part describes security tasks for Oracle API Catalog.

This part includes the following chapters:

- Chapter 7, "Configuring Oracle API Catalog to use External Authentication Tooling"
- Chapter 8, "Password Encryption"
This chapter describes how to authenticate Oracle API Catalog using the external authentication tools, such as LDAP / Active Directory.

This chapter contains the following sections:

- Section 7.1, "Overview"

7.1 Overview

This section describes how to use LDAP/Active Directory to authenticate the users in Oracle API Catalog.

This section contains the following topics:

- Section 7.1.1, "LDAP/Active Directory"
- Section 7.1.2, "Enable LDAP Integration System Properties"
- Section 7.1.3, "Modify LDAP/Active Directory Properties"
- Section 7.1.4, "Security Considerations"
- Section 7.1.5, "LDAP Property Examples"

7.1.1 LDAP/Active Directory

When utilizing LDAP / Active Directory (AD) to authenticate users, some consideration must be given to the user’s department and role configurations prior to the utilization of LDAP / Active Directory server. All users are authenticated through LDAP/AD once the integration is enabled; it is essential to have at least one admin-level user account created within the Oracle API Catalog database that matches the username from LDAP/AD. This user account should be assigned the admin role so that administrative functions within Oracle API Catalog can still be performed when LDAP/AD is enabled.

If role synchronization is enabled from LDAP/AD, at least one user account should be assigned an administrative-level role. LDAP roles should be created and appropriate Oracle API Catalog permissions assigned within Oracle API Catalog prior to utilizing the role-synch option. Role names are synchronized on name only. When the administrative-level user logs into Oracle API Catalog using LDAP/AD, that person has the ability to configure and administer the application properly. This user account should NOT be used for daily activities.
7.1.2 Enable LDAP Integration System Properties

This procedure is performed on the Oracle API Catalog Admin screen.

1. Click System Settings in the left pane. The System Settings section opens in the main pane.
2. Use the System Settings Search box to easily locate each of the following settings.
3. Enter `enterprise.authentication.ldap.enabled` into the Search box. Set the value to `True` and click Save.

Change the settings as indicated below:

- Unapproved User Login
  - Set to `True`.
- Plug-in Login Module Class Name
  - Enter `com.flashline.enterprise.authentication.server.loginmodule.LDAPLogin` in the text box.

Note: This property turns LDAP on/off. Once enabled, the application uses LDAP server for user authentication.

4. Click Save when finished.

7.1.3 Modify LDAP/Active Directory Properties

You can modify LDAP/AD properties, as follows:

1. Click System Settings in the left pane.
2. Use the System Settings Search to easily locate each of the following settings. Enter the values as indicated below:

   - LDAP Server Host Name
     - In the text box, enter the Host name, or the directory server IP address.
   - LDAP Server Port Number
     - Enter `389` in the text box.
   - LDAP Mask
     - Enter `uid=^` for LDAP
     - Enter `samAccountName=^` for Active Directory
   - Creation of Unapproved User Accounts
     - Set to `True`.
   - Assign default roles to users
     - Set to `True`.

Note: This property assigns default roles on every user authentication.
- Auto create missing roles
  - Set to True.

**Note:** This property creates role synchronized from the LDAP/AD server, but does NOT assign any permissions to those roles.

- Auto create missing departments
  - Set to True.

**Note:** This property creates departments synchronized from the LDAP/AD server, but does not assign any description to those departments. However, the user is assigned to the new role.

- LDAP Version
  - Enter 3 in the text box. (Supported versions are 2 and 3)

- Administrator Account Distinguished Name

**Note:** This property is required with using Active Directory. This property must contain a DN of a user account with at least read-only directory look-up permissions.

Example: CN=Some_User,CN=Users,DC=ad,DC=example,DC=com

- Administrator Account Password
  - In the text box, enter the password for the administrator account identified in the Administrator Account Distinguished Name property, above.

- Use SSL Connection
  - Set to True to enable an SSL connection for LDAP. The default value is false.

- Follow referrals
  - Set to True.

- Retrieve data using the admin account
  - Set to False for LDAP (if applicable) Or
  - Set to True for Active Directory or restricted LDAP environments.

- Search Start Location

**Note:** This property defines where in the directory tree the search for user records begins.

Examples:
- For LDAP: OU=MemberGroupB, O=en_us
- For Active Directory: CN=Users,DC=ad,DC=example,DC=com

- Search Scope
- Select subtree in the list.

**Note:** This property defines the depth (below the baseDN) of user record searches.

- Attribute Name that Identifies a Found Entry

  **Note:** This property designates the attribute name that uniquely identifies the user account within the scope of the tree search.

  - For LDAP: uid
    - Or
    - For Active Directory: samAccountName

- Found Entry Email Attribute Name
  - Enter mail

- Found Entry First Name Attribute Name
  - Enter givenName

- Found Entry Middle Name Attribute Name
  - Enter the middle name attribute from your LDAP or Active Directory (if applicable)

- Found Entry Last Name Attribute Name
  - Enter sn

- Found Entry Telephone Number Attribute Name
  - Enter telephoneNumber

- Use LDAP Departments
  - Set to True

  **Note:** This property defines the user's department attribute value that is synchronized within Oracle API Catalog.

- Department Attribute
  - Enter department

- Use LDAP Roles
  - Set to False

- Role Attribute
  - Enter the LDAP / Active directory attribute that contains the role information for the user.

- Second Level Lookup Attribute
3. Click **Save** when finished.

4. Restart the Oracle API Catalog application.

### 7.1.4 Security Considerations

Using the Oracle API Catalog LDAP/Active Directory Connector allows LDAP to act as the single source of user identification for Oracle API Catalog user authentication and role assignment. However, this does not prevent respective host repositories from managing user authentication for access to files through Oracle API Catalog.

When using the Oracle API Catalog LDAP/Active Directory Connector, Oracle API Catalog depends on LDAP or Active Directory to authenticate users. The username/password combination is delegated to the LDAP system as a bind request. The user is authenticated only if the bind request is successful.

As an option, LDAP can be configured to store/retrieve Oracle API Catalog user role assignments. In this configuration, at each user login Oracle API Catalog synchronizes with the user's roles as stored in LDAP. Roles are added directly through LDAP, and are not managed by Oracle API Catalog; the group mappings within the LDAP are not used by OER. To use this option, the `ldap.enable-synch-roles` system settings must be set to `True`. The `ldap.rbac.roleAttrib` system setting must be set to the name of the multi-value LDAP attribute that contains the role names that the user has within Oracle API Catalog. If the `ldap.enable-synch-roles` system setting is not enabled, or if the LDAP attribute referenced by the `ldap.rbac.roleAttrib` option is not configured, an administrator manually assigns user roles in Oracle API Catalog.

#### 7.1.4.1 Use Case Sample Scenarios

The following scenarios illustrate a selection of LDAP setups and configurations in order to clarify property settings for user management.

**Scenario 1**

Prevent user access to Oracle API Catalog despite LDAP authentication. Access is provided only to pre-existing users with active Oracle API Catalog accounts.

- **Rationale**
  - Non-enterprise license agreements where user base is predefined and number of users allowed into the application is limited.

- **Property Settings**
  - `ldap.allow-user-creation`
    * Set to **False**
  - `enterprise.security.unapproveduser.allowlogin`
    * Set to **False**
**Scenario 2**
On LDAP authentication, create a default Oracle API Catalog user account and assign the default role(s), but deny the user access to the Oracle API Catalog.

In Oracle API Catalog, the default role is "developer".

- **Rationale**
  - To deny Oracle API Catalog access to a new user until the security administrator is notified that the new user account was created. Once approved by the security administrator, the user's status is changed to active, allowing Oracle API Catalog login.

- **Property Settings**
  - `ldap.allow-user-creation`
    - * Set to True
  - `ldap.assign-default-roles`
    - * Set to True
  - `enterprise.security.unapproveduser.allowlogin`
    - * Set to False

**Scenario 3**
On LDAP authentication, a default Oracle API Catalog user account is created with the default role(s), and the user is permitted to login to the Oracle API Catalog.

- **Rationale**
  - An enterprise license agreement in which LDAP authentication is the only restriction on new user creation. Typically, the default Oracle Enterprise Repository role would be set to developer in order to limit access for new users whose roles are not predefined by an LDAP account.

- **Property Settings**
  - `ldap.allow-user-creation`
    - * Set to True
  - `ldap.assign-default-roles`
    - * Set to True
  - `enterprise.security.unapproveduser.allowlogin`
    - * Set to True

### 7.1.5 LDAP Property Examples

Since limitations in Active Directory prevent searches below the top-level of the directory while anonymously bound (not authenticated) to the directory server, Oracle API Catalog user information lookup requires the Bind DN, Bind Password, and Retrieve Data As Admin properties to be set with appropriate values. Table 7–1 describes the LDAP / Active Directory properties with its examples.
Overview
Configuring Oracle API Catalog to use External Authentication Tooling

Table 7–1 Active Directory and Traditional LDAP Properties

<table>
<thead>
<tr>
<th>Active Directory</th>
<th>Traditional LDAP (InetOrgPerson)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldap.host</td>
<td>ad.example.com</td>
</tr>
<tr>
<td>ldap.port</td>
<td>389</td>
</tr>
<tr>
<td>ldap.version</td>
<td>3</td>
</tr>
<tr>
<td>ldap.bindDN</td>
<td>CN=Some_User,OU=Users,DC=ad,DC=example,DC=com</td>
</tr>
<tr>
<td>ldap.bindPassword</td>
<td>password</td>
</tr>
<tr>
<td>ldap.retrieve-data-as-admin</td>
<td>true</td>
</tr>
<tr>
<td>ldap.mask</td>
<td>sAMAccountName=^</td>
</tr>
<tr>
<td>ldap.baseDN</td>
<td>CN=Users,DC=ad,DC=example,DC=com</td>
</tr>
<tr>
<td>ldap.scope</td>
<td>subtree</td>
</tr>
<tr>
<td>ldap.uniqueIDAttrib</td>
<td>samAccountName</td>
</tr>
<tr>
<td>ldap.emailAttrib</td>
<td>mail</td>
</tr>
<tr>
<td>ldap.givennameAttrib</td>
<td>givenname</td>
</tr>
<tr>
<td>ldap.surnameAttrib</td>
<td>sn</td>
</tr>
<tr>
<td>ldap.telephoneAttrib</td>
<td>telephonenumber</td>
</tr>
<tr>
<td>ldap.deptAttrib</td>
<td>department</td>
</tr>
</tbody>
</table>

Table 7–2 Custom and Common LDAP Properties

<table>
<thead>
<tr>
<th>Custom and Common Properties Regardless of implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldap.deptAttrib</td>
</tr>
<tr>
<td>ldap.rbac.roleAttrib</td>
</tr>
<tr>
<td>ldap.allow-user-creation</td>
</tr>
<tr>
<td>ldap.enable-synch-roles</td>
</tr>
<tr>
<td>ldap.enable-synch-depts</td>
</tr>
</tbody>
</table>
This chapter provides information about how you can encrypt passwords.

Password encryption is enabled by default in Oracle API Catalog.

This chapter contains the following sections:

- Section 8.1, "Generating Encrypted Passwords"
- Section 8.2, "Encrypting the Configuration File Passwords"

### 8.1 Generating Encrypted Passwords

You can encrypt passwords using two different methods, one using the Oracle API Catalog diagnostic page and other using the password encryption tool. To generate encrypted passwords, you need to perform the following steps:

1. Access the Oracle API Catalog Diagnostics page.
   
   Navigate to http://host_name:port/application_name/diag/index.jsp (replace host_name with the appropriate location).

2. Scroll down to the Tools section and click the Encrypt Strings for passwords link to launch the Password encryption page.

3. Enter the clear text password into the String to Encrypt text box.

4. Click the Submit Query button.

5. Copy the resulting encrypted password string and paste it into the appropriate context or properties file(s).

---

**Note:** In Oracle API Catalog 12c release, Oracle API Catalog Diagnostics page is disabled by default.

To see if the diagnostics page is currently disabled, navigate to http://host_name:port/application_name/diag/index.jsp (replace host_name with the appropriate location). If the diagnostics page is disabled, the following message is displayed:

*Diag pages are currently disabled. Please contact your Oracle Enterprise Repository.*

See Section 2.6, "Accessing Oracle API Catalog Diagnostics Page" for information about enabling the diagnostics page.
The password encryption tool can be found at `<FMW_HOME>/oer/modules/tools/solutions/12.1.3.0.0-OER-PasswordTools.zip`. The 12.1.3.0.0-OER-PasswordTools.zip file has two scripts (for Windows and Unix):

- `encrypt.bat/encrypt.sh` - encrypt an xml config file’s password elements
- `encryptpassword.bat/encryptpassword.sh` - encrypt a single password from the command line

This section also contains the following topics:

- Section 8.1.1, "Encrypted Passwords Used in Oracle API Catalog Deployment"
- Section 8.1.2, "Other Passwords"

### 8.1.1 Encrypted Passwords Used in Oracle API Catalog Deployment

The suggested usage of encrypted passwords are as below:

- In the `database.properties` file
  The connection password for the database.

- The Ant task property file or build script
  The password the Oracle API Catalog user will use at login.

- In the Harvester `HarvesterSettings.xml` configuration file
  The password stored in the `HarvesterSettings.xml` file.

### 8.1.2 Other Passwords

Other passwords in the system are encrypted automatically. This operation is invisible to the user. A number of fields stored in the properties files are encrypted by default, including:

- `ldap.bindPassword`
- `enterprise.guest.password`

This encryption occurs when the properties are edited and saved. Automatic encryption of passwords during an upgrade is unavailable at this time.

Passwords stored with the artifact stores are stored in the database in an encrypted format.

### 8.2 Encrypting the Configuration File Passwords

To ensure security, the passwords in the configuration files must be encrypted. You need to encrypt the configuration file passwords for the following:

- Section 8.2.1, "Harvester Configuration File"

### 8.2.1 Harvester Configuration File

To ensure security, the passwords in the harvester configuration must be encrypted. The password encryption tool, (`encrypt.bat/encrypt.sh`), which is found at `oer/modules/tools/solutions/12.1.3.0.0-OER-PasswordTools.zip`, allows you to encrypt the passwords that are stored in the Harvester configuration (`HarvesterSettings.xml`) file.

1. Navigate to the `<Harvester Home>` directory.
2. From a command prompt, run the password encryption tool as follows:

> encrypt.bat HarvesterSettings.xml HarvesterSettings.xml

where

HarvesterSettings.xml = the Harvester configuration file.