Oracle® Identity Manager

Connector Guide for Generic SCIM

Release 11.1.1
E72359-05
June 2020

Oracle Identity Manager Connector Guide for Generic SCIM, Release 11.1.1

E72359-05

Primary Author: Alankrita Prakash

Contributing Authors: Gowri.G.R

Contents

2 Generating the Generic SCIM Connector

3 Installing and Configuring the Generic SCIM Connector

List of Figures

List of Tables

Preface

This guide describes the connector that is used to integrate Oracle Identity Manager with SCIM-based target systems.

Audience

This guide is intended for resource administrators and target system integration teams.

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 information about installing and using Oracle Identity Manager, visit the following Oracle Help Center page:

http://docs.oracle.com/cd/E52734_01/index.html

For information about Oracle Identity Manager Connectors documentation, visit the following Oracle Help Center page:

http://docs.oracle.com/cd/E22999_01/index.htm

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.

Convention	Meaning
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

What's New in Oracle Identity Manager Connector for Generic SCIM Connector?

This chapter provides an overview of the updates made to the software and documentation for the Generic SCIM connector in release 11.1.1.5.0.

The updates discussed in this chapter are divided into the following categories:

- Software Updates

  This section describes updates made to the connector software. This section also points out the sections of this guide that have been changed in response to each software update.

- Documentation-Specific Updates

  These include major changes made to this guide. For example, the relocation of a section from the second chapter to the third chapter is a documentation-specific update. These changes are not related to software updates.

Software Updates

The following section discusses software updates:

Software Updates in Release 11.1.1.5.0

This is the first release of the Oracle Identity Manager connector for Generic SCIM. Therefore, there are no software-specific updates in this release.

Documentation-Specific Updates

The following section discusses documentation-specific updates:

Documentation-Specific Updates in Release 11.1.1.5.0

The following are documentation-specific updates in revision "05" of this guide:

- The "Oracle Identity Governance or Oracle Identity Manager" row of Certified Components for Generic SCIM Connector has been updated to include support for Oracle Identity Governance release 12*c* PS4 (12.2.1.4.0).
- A Note about using Oracle Identity Governance release 12*c* PS4 (12.2.1.4.0) as a SCIM target has been added under the customHeaders row of IT Resource Parameters .

The following is a documentation-specific update in revision "04" of this guide:

The "customHeaders" and "httpOperationTypes" rows in Table 3-8 have been updated.

The following is the documentation-specific update in revision "03" of this guide:

Additional certification details for Oracle Identity Governance 12c (12.2.1.3.0) added to Certified Components for Generic SCIM Connector.

The following are the documentation-specific updates in revision "02" of this guide:

- The sample value for the entitlementAttributeList entry has been modified from `__ACCOUNT__.groups.value` to `__ACCOUNT__.groups~__ACCOUNT__.groups~value` in Understanding Entries in the Predefined Sections of the Groovy File.
- The "Object Type" row of Table 4-5 has been modified to include a note.

The following is the documentation-specific update in revision "01" of this guide:

This is the first release of the Oracle Identity Manager connector for Generic SCIM. Therefore, there are no document-specific updates in this release.

1

About the Generic SCIM Connector

The Oracle Identity Manager Connector for Generic SCIM (Generic SCIM connector) integrates Oracle Identity Manager with SCIM -based target systems.

Oracle Identity Manager is a centralized identity management solution that provides self service, compliance, provisioning and password management services for applications residing on-premise or on the Cloud. Oracle Identity Manager connects users to resources, and revokes and restricts unauthorized access to protect sensitive corporate information. Oracle Identity Manager connectors are used to integrate Oracle Identity Manager with external and identity-aware applications such as PeopleSoft and MySQL.

The following topics introduce the Generic SCIM connector:

1.1 Introduction to the Generic SCIM Connector

The Generic SCIM connector is a solution to integrate Oracle Identity Manager with SCIM-based identity-aware applications. A SCIM-based identity-aware application is any application that exposes its SCIM APIs or interfaces for identity management.

> **Note:**
>
> In this guide:
>
> - A SCIM-based identity-aware application has been referred to as the **target system** or **SCIM-based target system.**
> - *RELEASE_NUMBER* has been used as a placeholder for the current release number of the connector. Therefore, replace all instances of *RELEASE_NUMBER* with the release number of the connector. For example, 11.1.1.
> - The Oracle Identity Manager Connector for Generic SCIM has been referred to as the **Generic SCIM connector**.

The Generic SCIM connector provides a centralized system to streamline delivery of services and assets to your company's consumers, and manage those services and assets in a simple, secure, and cost efficient manner by using automation. The Generic SCIM connector standardizes service processes and implements automation to replace manual tasks.

In order to connect with a SCIM-based target system, the Generic SCIM connector supports HTTP Basic Authentication and OAuth 2.0 authentication mechanisms. This connector also supports authenticating to the target system by using access token and refresh token as an input from the user. This authentication mechanism can be useful if your target system does not provide a programmatic approach to obtain access or refresh tokens.

The connector supports the following OAuth 2.0 grant types:

- JWT
- Client Credentials
- Resource Owner Password

If your target system does not support any of the authentication types supported by this connector, then you can implement the custom authentication that your target system supports. You can connect this custom implementation to the connector by using the plug-ins exposed by this connector.

The Generic SCIM connector synchronizes data between Oracle Identity Manager and SCIM-based target systems by performing reconciliation and provisioning operations that parse data in the JSON format. If your target system does not support request or response payload in JSON format, then you can create your own implementation for parsing data. You can connect this custom implementation to the connector by using the plug-ins exposed by this connector.

The Generic SCIM connector is a connector for a discovered target system. This is because the schema of the SCIM-based target system with which the connector integrates is not known in advance. The Generic SCIM connector is not shipped with any artifacts. Instead, it is shipped with a set of deployment utilities that help in discovering the schema of the SCIM-based target system and generating the artifacts.

1.2 Certified Components for Generic SCIM Connector

These are the software components and their versions required for integrating Oracle Identity Manager with a Generic SCIM connector.

Table 1-1 lists the certified components for this connector:

Table 1-1 Certified Components

Item	Requirement
Oracle Identity Manager	You can use one of the following releases of Oracle Identity Manager: • Oracle Identity Governance 12*c* (12.2.1.4.0) • Oracle Identity Governance 12*c* (12.2.1.3.0) • Oracle Identity Manager 11*g* Release 2 PS3 (11.1.2.3.0) • Oracle Identity Manager 11*g* Release 2 PS2 (11.1.2.2.0)
Target System	Any target system that supports SCIM-based services.
Connector Server	• 11.1.2.1.0 • 12.2.1.3.0
Connector Server JDK	JDK 1.6 or later

1.3 Certified Languages for the Generic SCIM Connector

These are the languages that the connector supports.

- Arabic
- Chinese (Simplified)
- Chinese (Traditional)
- Czech
- Danish
- Dutch
- English (US)
- Finnish
- French
- French (Canadian)
- German
- Greek
- Hebrew
- Hungarian
- Italian
- Japanese
- Korean
- Norwegian
- Polish

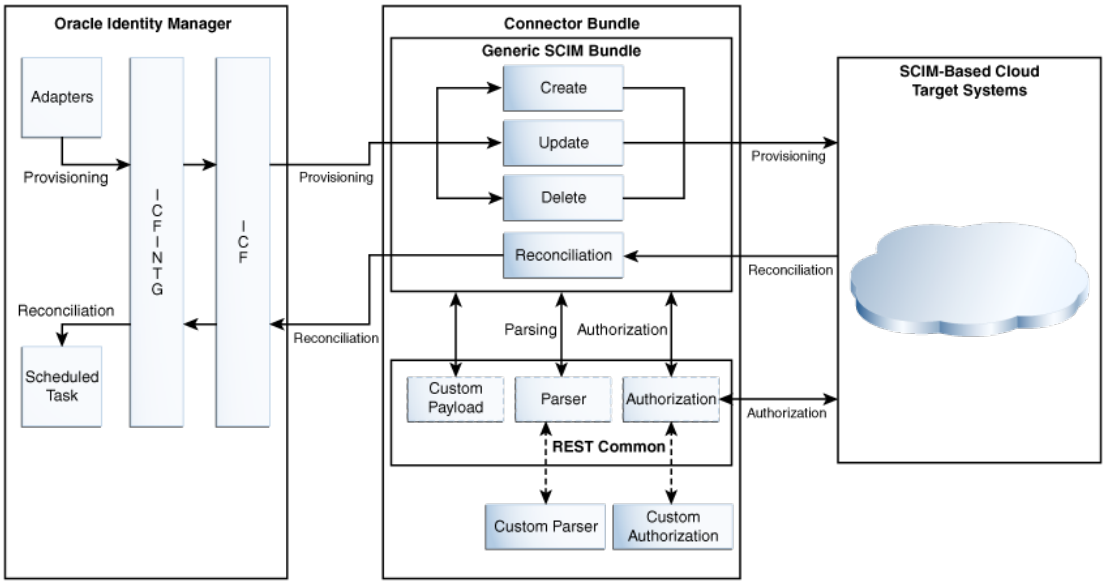
- Portuguese
- Portuguese (Brazilian)
- Romanian
- Russian
- Slovak
- Spanish
- Swedish
- Thai
- Turkish

1.4 Architecture of the Generic SCIM Connector

The Generic SCIM connector is implemented using the Identity Connector Framework (ICF).

The ICF is a component that provides basic reconciliation and provisioning operations that are common to all Oracle Identity Manager connectors. In addition, ICF provides common features that developers would otherwise need to implement on their own, such as connection pooling, buffering, time outs, and filtering. The ICF is shipped along with Oracle Identity Manager.

Figure 1-1 Generic SCIM Connector Architecture



The Generic SCIM connector is not shipped with any metadata as it is a connector for target system that is not known in advance. Depending on the schema of your target system, the connector artifacts are generated during connector deployment.

The following is a high-level description of the stages into which the connector deployment and usage procedure is divided into:

- **Generating the Connector**

  The Generic SCIM connector includes a groovy file in which you can specify information about your target system. This information is used by the metadata generator, one of the deployment utilities shipped with the connector, to generate the connector based on the target system schema.

  When you run the metadata generator on the groovy file, the connector package is generated. This package contains an XML file that contains definitions for connector components such as adapters, process tasks, scheduled tasks, lookup definitions, and IT resource. Connector operations such as provisioning and reconciliation are performed using these connector components. Along with the XML file, a schema file is included.

- **Installing and configuring the connector**

  In this stage, you install the generated connector by running the connector installer and then perform configuration tasks such as configuring the IT resource, enabling logging and so on.

- **Using the Connector**

  In this stage, you start using the connector to perform connector operations such as reconciliation and provisioning.

1.5 Use Cases Supported by the Generic SCIM Connector

The Generic SCIM connector can be used to integrate OIM with any target system that supports SCIM services. This connector can be used to load identity data into OIM from a SCIM service and then efficiently manage identities in an integrated cycle with the rest of the identity-aware applications in your enterprise.

Oracle Identity Manager Connector for Generic SCIM, with a few simple configurations, provides a reusable framework that helps in integrating most of the SCIM-based target systems. This connector can be used to load identity data into Oracle Identity Manager from a SCIM service and then efficiently manage identities in an integrated cycle with the rest of the identity-aware applications in your enterprise.

As a business use case example, consider a leading logistics company that has 100+ cloud applications. Most of these cloud applications are now inefficient because data in these applications are manually entered and are managed using spreadsheets or custom-coded process flows. Therefore, this company wants to integrate its cloud applications with Oracle Identity Manager to streamline its operations, increase its organizational efficiency, and at the same time, lower its operational costs. There are two approaches for integrating these cloud applications with Oracle Identity Manager . One approach would be to deploy a point-to-point connector for each of these applications. The drawbacks of this approach are as follows:

- Increased time and effort to identify and deploy a point-to-point connector for each application.
- Increased administration and maintenance overheads for managing connectors for each application.
- Unavailability of point-to-point connectors for all applications. In such a scenario, one needs to develop custom connectors which increases time and effort to develop, deploy and test the custom connector.

An alternative to this approach is to use the Generic SCIM connector that can be used to integrate all the cloud applications with Oracle Identity Manager . The Generic

SCIM connector provides the ability to manage accounts across all cloud applications without spending additional resources and time on building custom connectors for each cloud application.

The Generic SCIM connector is a hybrid approach that helps enterprises leverage on-premise Oracle Identity Manager deployment to integrate with target systems for identity governance. These targets systems include any application that exposes SCIM APIs such as SaaS, PaaS, home-grown applications and so on.

The following are some example scenarios in which the Generic SCIM connector is used:

- **User Management**

  The Generic SCIM Connector manages individuals who can access Cloud service by defining them as users in the system and assigning them to groups. This connector allows new users to self-provision on a Generic SCIM Cloud Service, while having it be controlled by IT. Users can request and provision from a catalog of cloud-based resources that is established by Oracle Identity Manager administrators. For example, to create a new user in the target system, fill in and submit the Oracle Identity Manager process form to trigger the provisioning operation. The connector executes the create operation against your target system and the user is created on successful execution of the operation. Similarly, operations such as delete and update can be performed.

- **Entitlement Management**

  The Generic SCIM Connector manages Cloud services objects (if exposed by the target system) as entitlements. Depending on the target system being used, this connector can be used to manage entitlements such as Groups, Roles, Licenses, Folders, Collaboration and so on. For example, you can use the Generic SCIM connector to automatically assign or revoke groups to users based on predefined access policies in Oracle Identity Manager . Similarly, you can use the Generic SCIM Connector to manage role memberships that provide selective access to certain Cloud Service functionality or groups. Therefore, as new users are added to a specific role, they automatically gain corresponding access in the applications. As an administrator, you can also use this connector to efficiently manage user licenses for all the available resources. By leveraging the auditing and reporting tools of Oracle Identity Manager, you can automate license allocation whenever a new account is created. In addition, license assignments and usage can be monitored through changing organization needs and unused licenses can be tracked for potential recycling.

1.6 Features of the Generic SCIM Connector

The features of the connector include support for full and incremental reconciliation, limited reconciliation, custom authentication, custom parsing, custom payload, handling multiple endpoint URLs, and SSL communication.

The following are the features of the connector:

- Support for Both Trusted Source and Target Resource Reconciliation
- Full and Incremental Reconciliation
- Limited (Filtered) Reconciliation
- Custom Authentication

- Custom Parsing
- Custom Payload
- Support for Additional HTTP Headers
- Support for Handling Multiple Endpoint URLs
- SSL Communication

1.6.1 Support for Both Trusted Source and Target Resource Reconciliation

The Generic SCIM connector includes a groovy file (a part of the metadata generator) that enables you to configure the connector to run either in the trusted source mode or target resource mode.

See Configuring the GenericScimConfiguration.groovy File.

1.6.2 Full and Incremental Reconciliation

After you create the connector, you can perform full reconciliation to bring all existing user data from the target system to Oracle Identity Manager. After the first full reconciliation run, you can configure your connector for incremental reconciliation. In incremental reconciliation, only records that are added or modified after the last reconciliation run are fetched into Oracle Identity Manager.

> **Note:**
>
> The connector supports incremental reconciliation if the target system contains an attribute that holds the timestamp at which an object is created or modified. See Full Reconciliation and Incremental Reconciliation.

You can perform a full reconciliation run at any time.

1.6.3 Limited (Filtered) Reconciliation

You can reconcile records from the target system based on a specified filter criterion. To limit or filter the records that are fetched into Oracle Identity Manager during a reconciliation run, you can specify the subset of added or modified target system records that must be reconciled.

You can set a reconciliation filter as the value of the Filter attribute of the scheduled jobs. This filter specifies the subset of newly added and modified target system records that must be reconciled. See Limited Reconciliation for Generic SCIM Connector.

1.6.4 Custom Authentication

By default, the Generic SCIM connector supports HTTP Basic Authentication and OAuth 2.0 authentication mechanisms. The connector also supports an authentication mechanism in which the user provides access token and refresh tokens as an input.

The supported grant types for OAuth 2.0 authentication mechanism are JWT, Client Credentials, and Resource Owner Password.

If your target system uses any of the authentication mechanisms that is not supported by the connector, then you can write your own implementation for custom authentication by using the plug-ins exposed by this connector. See Implementing Custom Authentication.

1.6.5 Custom Parsing

By default, the Generic SCIM connector supports request and response payloads only in the JSON format. If your target system does not support request or response payload in JSON format, then you can implement a custom parsing logic by using plug-ins exposed by this connector.

See Implementing Custom Parsing.

1.6.6 Custom Payload

The Generic SCIM connector provides support for handling custom formats for any attributes in the payload that do not adhere to the standard JSON format. This can be achieved by specifying a value for the customPayload IT resource parameter.

1.6.7 Support for Additional HTTP Headers

If your target system requires additional or custom HTTP headers in any SCIM call, then you can insert these HTTP headers as the value of the customAuthHeaders or customAuthHeaders IT resource parameters. See Additional Configuration Parameters..

1.6.8 Support for Handling Multiple Endpoint URLs

The Generic SCIM connector allows you to handle attributes of an object class (for example, a User object class) that can be managed only through endpoints other than the base endpoint URL of the object class. For example, in certain target systems, there are attributes of the User object class that can be managed using the base endpoint URL. However, some attributes (for example, email alias) can be managed only through a different endpoint URL. The connector provides support for handling all endpoint URLs associated with an object class.

This can be achieved by providing endpoint URL details of such attributes in the relURIs IT resource parameter.

1.6.9 SSL Communication

You can configure SSL communication between Oracle Identity Manager and the SCIM-based target system.

See Configuring SSL for the Generic SCIM Connector for information about configuring secure communication.

1.7 Roadmap for Generating and Using the Connector

This is the organization of information available in this guide for deploying and using the connector.

The rest of this guide is divided into the following chapters:

- Generating the Generic SCIM Connector describes the procedure that you must perform to configure the groovy file and to run the metadata generator to generate the connector.
- Installing and Configuring the Generic SCIM Connector describes that procedures that you must perform during each stage of connector installation.
- Using the Generic SCIM Connector describes guidelines on using the connector and the procedure to configure reconciliation runs and perform provisioning operations.
- Extending the Functionality of the Generic SCIM Connector describes procedures that you can perform if you want to extend the functionality of the connector.
- Files and Directories of the Generic SCIM Connector lists the files and directories that comprise the connector installation media.

2
Generating the Generic SCIM Connector

The procedure to generate the Generic SCIM connector is divided across three stages namely defining the schema, configuring the Groovy file, and generating the connector.

- Defining the Schema
- Configuring the GenericScimConfiguration.groovy File
- Generating the Generic SCIM Connector

2.1 Defining the Schema

You can define the schema of your target system to let the connector understand the underlying schema of the target system database.

The following topics are discussed in this section:

> **Note:**
>
> Defining a schema is optional. When schema is not defined, the Generic SCIM connector fetches all attributes that are returned by the endpoint of the target system.

- Understanding the Schema File Format
- Creating a Schema File

2.1.1 Understanding the Schema File Format

The schema file is a properties file that is used to represent the structure of your target system. This file is used as an input to the metadata generation utility. Before running the metadata generation utility, you must populate the schema file in the specified format.

> **Note:**
>
> Creating a schema is optional. When a custom schema is not available, the Generic SCIM connector fetches all attributes that are returned by the endpoint of the target system. Properties of the fields mentioned in the schema file take a higher preference over the properties returned by the schema endpoint of the Generic SCIM connector

The schema file contains details such as datatypes, mandatory attributes, and the uid attribute that are specific to your target system. This file restricts the list of field names

which are managed from the Generic SCIM connector. It is necessary to create a schema.properties file to help the connector understand the target system schema.

The schema file is a properties file and consists of name-value pairs. By default, the metadata generation utility generates metadata for an __ACCOUNT__ object class that is used to manage Users, groups, and organizations.

The formats in which you must specify the value of each property are discussed in the following sections:

- Account Qualifiers
- Field Qualifiers

2.1.1.1 Account Qualifiers

Account qualifiers describe certain attributes of an account in your target system. These qualifiers are common for the target system.

You can define the schema of your target system by using the following qualifiers:

- **Field Names**

  This is a mandatory qualifier. It is a comma-separated list of attributes that the connector must fetch from the target system. All child form names, single-valued and multivalued attributes, including the attribute used for performing incremental reconciliation must be specified here.

  The following is a sample value for the FieldNames qualifier:

  ```
  FieldNames=UserId,UserName,FirstName,LastName,email,Description,Salary
  , JoiningDate,status,Groups,Roles
  ```

- **UidAttribute**

  This is a mandatory qualifier. It refers to the name of the attribute that corresponds to the unique ID of the account.

  For example: `UidAttribute=UserId`

- **NameAttribute**

  This is a mandatory qualifier. This refers to the name of the attribute that corresponds to a descriptive name of the account.

  For example: `NameAttribute=UserName`

- **PasswordAttribute**

  This is a mandatory qualifier if your target system supports password management of user accounts. This qualifier refers to the name of the password attribute of the account.

  For example: `PasswordAttribute=accountPwd`

- **StatusAttribute**

  This is a mandatory qualifier if your target system supports management of user account statuses. This qualifier refers to the attribute which denotes the status of the account.

  For example: `StatusAttribute=status`

2.1.1.2 Field Qualifiers

Field qualifiers are specific to each field and are usually specified by predefined formats.

- The following is the format for parent form fields:

  <FIELDNAME>.<FIELDQUALIFIER>=<VALUE>

  Example: `UserId.Required=true`

- The following is the format for complex child form fields:

  <FIELDNAME>.<SUBFIELDNAME>.<FIELDQUALIFIER>=<VALUE>.

  Example: `Roles.fromdate.DataType=Long`

The following are the field qualifiers for which values can be specified:

- **Required**

  This field qualifier specifies if the mentioned attribute is mandatory. If the value of this qualifier is set to true, the parser will skip processing the records that do not contain this field name.

  For example: `UserId.Required=true`

- **Multivalued**

  This field qualifier specifies if the mentioned attribute is a multivalued field.

  For example: `Roles.Multivalued=true`

- **DataType**

  This field qualifier is used to specify the datatype of the field name. If you do not specify the data type for any field, then it is considered as a String data type by default.

  The following are the possible values for this qualifier:

  - String
  - Long
  - Character
  - Double
  - Float
  - Integer
  - Boolean
  - Byte
  - BigDecimal
  - bigInteget
  - Date

  For example: `startDate.DataType=Date`

- **Subfields**

This field qualifier specifies the subfields in a multivalued attribute if they are present.

For example: `Roles.Subfields=roleid,fromdate,todate`

- **EmbeddedObjectClass**

  This field qualifier specifies the object class name of child forms that have more than one subfield. The value of this qualifier is used internally by ICF and is mandatory for all complex child forms.

  For example: `Roles.EmbeddedObjectClass=Roles`

2.1.2 Creating a Schema File

You can create a schema file for your target system by describing the structure, by adding mandatory qualifier entries and lastly by providing values for the newly created entries.

> **Note:**
>
> You must create the schema.properties file on the computer on which you intend to run the metadata generation utility.

1. Create a .properties file.
2. Add entries in the schema file according to requirements of your environment.

   The following are the mandatory qualifiers that should be defined in the schema file:

   - `FieldNames`
   - `UidAttribute`
   - `NameAttribute`
3. Provide values for each of the entries that you added. See Understanding the Schema File Format for more information about the format in which you must specify these values.
4. Save the created .properties file.

2.2 Configuring the GenericScimConfiguration.groovy File

The Generic SCIM connector is shipped with a groovy file named GenericScimConfiguration.groovy.

- About the GenericScimConfiguration.groovy File
- Understanding Entries in the Predefined Sections of the Groovy File
- Updating the Groovy File

2.2.1 About the GenericScimConfiguration.groovy File

This GenericScimConfiguration.groovy file is located in the genericscim-*RELEASE_NUMBER*/metadata-generator/resources directory of the connector

installation ZIP. You use the GenericScimConfiguration.groovy file to specify values for properties that can store basic information about your target system schema.

This file is used by the Generic SCIM Generator to perform the following tasks:

- Understand the schema
- Configure the mode (trusted source or target resource) in which you want to run the connector
- Generate the connector package specific to your target system

The procedure for running the Generic SCIM Generator and directory structure of the generated connector package is discussed later in this chapter.

The GenericScimConfiguration.groovy file contains sample configuration (one each for trusted source and target resource) with prepopulated values for most of the entries. Depending upon your requirements, specify or modify values for entries in this file or create new sections for your configuration. The following are the predefined sections in the GenericScimConfiguration.groovy file:

- trusted

  You specify values for the entries in this section if you want to configure the connector for the trusted source mode.

- target

  You specify values for the entries in this section if you want to configure the connector for the target resource mode.

2.2.2 Understanding Entries in the Predefined Sections of the Groovy File

The entries in the predefined sections, trusted and target, of the GenericScimConfiguration.groovy file are defined in this section.

> **Note:**
>
> - Unless specified, all entries described here are common to both sections.
> - If you do not want to specify a value for any of the optional entries or attributes in the GenericScimConfiguration.groovy file, then comment out that entry or attribute by prefixing it with the double-slash symbol (//).

The following are a list of entries:

- itResourceDefName

  This is a mandatory entry. Enter the name of the IT resource type for the target system. Note that the value that you specify for this entry determines the name of the connector package, connector configuration file, and connector installer file. For example, if you specify GenSCIMTrusted as the value of this entry, then the name of the connector package directory is GenSCIMTrusted.zip.

- itResourceName

This is an optional entry. Enter the name of the IT resource for the target system. If this entry is commented, then the IT resource name will be the same as the value of the ITResourceDefName entry.

Default value: `"$itResourceDefName"`

> **Note:**
>
> The value of this entry must be unique for each connector that you create for your target system, if you plan to install or use the connectors in the same Oracle Identity Manager environment. In addition, this value will be a part of the names for all connector components (defined in the connector configuration XML file, which is created after you run the metadata generator) such as lookup definitions, resource objects, process forms, and scheduled tasks.
>
> For example, if you specify `GenSCIMTrusted` as the value of itResourceName entry, then after you deploy the connector, the configuration lookup definition is created and its name will be Lookup.GenSCIMTrusted.Configuration.

- applicationInstanceName

  This is an optional entry and present only in the section for target resource configuration. Enter the name of the application instance for your target system that the connector must generate. If this entry is commented, then the application instance name will be the same as the value of the ITResourceDefName entry.

  Default value: `"$itResourceDefName"`

- connectorDir

  This is an optional entry. This entry is the complete path to the directory that must contain the connector package that is generated when you run the metadata generator. By default, the name of the directory containing the generated connector package is the same as the value of the itResourceDefName entry.

  Sample value: `"/scratch/jdoe/OIMPS3/mw4318/idm7854/server/ConnectorDefaultDirectory/GenSCIMTrusted"`

- xmlFile

  This is an optional entry. Enter the name and relative path of the XML file that must contain definitions of the connector objects. If you do not specify a value for this entry, then the file name is generated in the following format:

  IT_RES_DEF_NAME-ConnectorConfig.xml

  In this format, *IT_RES_DEF_NAME* is the value of the itResourceDefName entry.

  For example, if you have not specified a value for this entry and `GenSCIM` is the value of the itResourceDefName entry, then the name of the XML file that is generated is `GenSCIM-ConnectorConfig.xml`.

> **Note:**
>
> To easily identify files of a specific target system installation, it is recommended that the names of this generated XML file be prefixed with the name of the IT resource for the target system.

Sample value: `GenSCIM-ConnectorConfig.xml`

- configFileName

  This is an optional entry. Enter the name and relative path of the XML file that contains the configuration information of the connector objects. If you do not specify a value for this entry, then the file name is generated in the following format:

  IT_RES_DEF_NAME-CI.xml

  In this format, *IT_RES_DEF_NAME* is the value of the itResourceDefName entry.

  For example, if you have not specified a value for this entry and `GenSCIM` is the value of the itResourceDefName entry, then the name of the XML file that is generated is `GenSCIM-CI.xml`.

- propertiesFile

  This is an optional entry. Enter the name and relative path of the .properties file which contains the resource bundle translations. If you do not specify a value for this entry, then the file name is generated in the following format:

  IT_RES_DEF_NAME-generator.properties

  In this format, *IT_RES_DEF_NAME* is the value of the itResourceDefName entry.

  For example, if you have not specified a value for this entry and `GenSCIMTarget` is the value of the itResourceDefName entry, then the name of the properties file that is generated is `GenSCIMTarget-generator.properties`.

- version

  This is an optional entry. Enter the release number of the connector.

  Sample value: `11.1.1.5.0`

- trusted

  This is a mandatory entry and present only in the section for trusted source configuration. Set the value of the entry to `true`, if you are configuring the connector to run in the trusted source mode.

- bundleJar

  This is a mandatory entry. Enter the name and relative path of the JAR file containing the ICF bundle that the metadata generator will use.

  Default value: `../lib/org.identityconnectors.genericSCIM-1.0.11150.jar`

  Do *not* change the value of this entry.

- config

  This is a mandatory entry in which you specify information about the connector configuration. This connector configuration contains information about the manner in which the connector must behave and connect to the target system.

Table 2-1 lists and describes the properties of the Config entry.

Note:

- By default, the config entry contains only the schemaFile property. Depending on the target system and the authentication mechanism being used, you can add any or all of the properties described in this table. All the config entry properties that you add to the groovy file are displayed as IT resource parameters. It is recommended that you add the properties in the groovy file and specify their values from the Manage IT resources page of the Identity System Administration. See Configuring the IT Resource for the Target System for more information about properties specific to your target system and its authentication mechanism.
- In this guide, an attribute in an object class that can be managed only through a separate rest endpoint rather than the same endpoint of the base object class has been referred to as a **special attribute**.

Table 2-1 Properties of the Config Entry

Property	Description
schemaFile	Enter the name and relative path of the schema file that you want to use. See Defining the Schema for information about the schema file that you created. Note: This attribute is an optional attribute.
host	Host name or IP address of the computer hosting the target system.
port	Port number at which the target system is listening

baseURI	Base URI refers to the base relative URL of the SCIM target system.
	For example, consider the URL:
	`http://host:port/hcmCoreSetupApi/scim/Users`
	In the preceding case, the baseURI is
	`/hcmCoreSetupApi/scim`

nameAttributes	Specifies the mapping between the _NAME_ connector attribute and the corresponding target system attribute for each object class that the connector handles. Format: *OBJ_CLASS.ATTR_NAME* **Note:** All values in this parameter must be comma separated.

Table 2-1 (Cont.) Properties of the Config Entry

Property	Description
uidAttributes	Enter the mapping between the _UID_ (GUID) connector attribute and target attribute for each object class that the connector handles. Format: `OBJ_CLASS.ATTR_NAME` **Note**: All values in this parameter must be comma separated.
statusAttributes	Enter the mapping between the _ENABLE_ connector attribute the target attribute that holds the status for each object class this connector handles. Format: *`OBJ_CLASS.ATTR_NAME`* **Note:** All values in this parameter must be comma separated.
grantType	Specifies the authorization grant used by your target system. The following are the supported grant types and the possible values for this property: – HTTP Basic Authentication — `basic` —OAuth2.0 JWT — `jwt` – OAuth 2.0 Client Credentials — `client_credentials` – OAuth 2.0 Resource Owner Password — `password` – If you have written your own custom implementation for authentication, then the value is `custom`.
contentType	This entry holds the content type expected by the target system in the header. The content type can be `application/json`.
acceptType	This entry holds the accept type expected by the target system in the header. The accept type can be `application/json`.
jsonResourcesTag	Specifies the json tag value that is used during reconciliation for parsing multiple entries in a single response payload.
scimVersion	This entry holds the SCIM version of the target system. The valid range for this attribute is `1` to `19`.
attrToOClassMapping	This entry is used to map an attribute of one object class with another object class. For example, if the groups attribute of the `__ACCOUNT__` object class must be mapped to the `__GROUP__` object class, then enter `__ACCOUNT__.groups=__GROUP__`. Sample value: `__ACCOUNT__.groups=__GROUP__`

- lookupAttributeList

This is an optional entry and is present only in the section for target resource configuration. Enter the list of attributes in your target system that must be handled as lookup fields.

The connector creates a lookup field for each of the attributes specified in this entry and associates it with the corresponding lookup fields on the Oracle Identity Manager User process form.

If you want to create a lookup field for a single-valued or multivalued field, then enter the value in the following format:

['*FIELD_NAME*']

In this format, replace *FIELD_NAME* with the name of the single or multivalued field.

If you want create a lookup field for a multivalued field that is embedded then, enter the value in the following format:

['*OBJ_CLASS*.*SUB_FIELD_NAME*']

In this format, replace:

- *OBJ_CLASS* with the EmbeddedObjectClass name for the child form as specified in the schema file.
- *SUB_FIELD_NAME* with the subfield name for the child form as specified in the schema file.

The sample value of this entry is:

```
['_ACCOUNT_.groups.value']
```

You can modify the default value to meet the requirements in your environment.

For each of the attributes listed in the lookupAttributeList entry, the connector creates a lookup definition and scheduled job in the following format:

- Lookup definition format:

  Lookup.${*IT_RES_NAME*}.${*FIELD_NAME*}

  This lookup definition holds the lookup values reconciled from the target system.

- Scheduled job format:

  IT_RES_NAME Target *FIELD_NAME* Lookup Reconciliation

  This scheduled job is used to load or reconcile lookup values from your target system. See Scheduled Job for Lookup Field Synchronization for more information about the attributes of the scheduled job for lookup reconciliation.

In both the formats, the connector replaces:

- *IT_RES_NAME* with the value of the itResourceDefName entry.
- *FIELD_NAME* with the name of the field for which the lookup field is created.

- entitlementAttributeList

  This is also an optional entry and is present only in the section for target resource configuration. Enter the list of fully qualified attributes in the target system that must be tagged as entitlements.

The connector creates a lookup field for each of the attributes specified in this entry, assigns the lookup fields to a process form, and adds all the required properties of entitlements.

If you want to tag entitlements for multivalued fields, then enter the value in the following format:

["*MULTIVALUED_FIELD_NAME*"]

If you want to tag entitlements for a multivalued field that is embedded, then enter the value in the following format:

["*OBJ_CLASS*.*SUB_FIELD_NAME*"]

In this format, replace:

- *OBJ_CLASS* with the EmbeddedObjectClass name for the child form as specified in the schema file.
- *SUB_FIELD_NAME* with the subfield name for the child form as specified in the schema file.

Sample value: `["__ACCOUNT__.groups~__ACCOUNT__.groups~value"]`

You can modify the default value based on your schema.

In this value, `groups.value` is an embedded object class field and `value` is a multivalued field.

- dateAttributeList

  This is an optional entry. Enter the list of attributes that must be handled as date on the process form. Ensure that the data type of the attributes listed here is set to Long in the schema file.

  The connector creates a date editor for each of the attributes specified in this entry.

  If you want to handle single-valued or multivalued fields as date, then enter the value in the following format:

  ["*FIELD_NAME*"]

  In this format, replace *FIELD_NAME* with the name of the single or multivalued field.

  If you want to handle an embedded multivalued field as date, then enter the value in the following format:

  ["*OBJ_CLASS*.*SUB_FIELD_NAME*"]

  In this format, replace:

  - *OBJ_CLASS* with the EmbeddedObjectClass name for the child form as specified in the schema file.
  - *SUB_FIELD_NAME* with the subfield name for the child form as specified in the schema file.

  Sample value: `["JoiningDate","Roles.fromdate","Roles.todate"]`

  In this value, `Roles.fromdate` and `Roles.todate` are embedded multivalued fields.

- objectClassAlias

  If you want to generate the metadata for an object class other than `__ACCOUNT__` and `_GROUP_`, then enter an alias for your object class.

- alias

  This is a mandatory entry. The metadata generator uses aliases to create relationships between the attributes in the target system and resource object field names in Oracle Identity Manager. In addition, the metadata generator uses aliases to shorten long database names to meet the character-length restrictions on form names and form field names in Oracle Identity Manager. Aliasing can be used on column name, form name, and form field name levels. Note that the target system attributes are represented as connector attributes.

  Depending on the type of configuration, specify values for one of the following sections:

  – For trusted source configuration

    In the trusted source configuration section, you use the alias entry to map connector attributes or target system attributes to the Oracle Identity Manager User form field names. The mappings that you specify here are used to populate entries in the Recon Attribute map lookup definition for trusted source reconciliation.

    Note that some of the Oracle Identity Manager User form field names do not have the same display name internally. For such fields, you must ensure that you map the connector attribute or target system attribute to the internal name rather than the display name. The following table lists the names of the Oracle Identity Manager User form display names and their corresponding internal names:

    Table 2-2 lists and describes the properties of the Alias entry.

Table 2-2 Alias

Display Name	Internal Name
Organization	Organization Name
Manager	Manager Login
E-mail	Email

The following is the default value of the alias entry:

```
['__NAME__':'User Login', 'LastName':'Last
Name','Organization':'Organization Name', 'Employee
Type':'Xellerate Type', 'Role':'Role']
```

In the default value, note that the `Organization` connector attribute has been mapped to `Organization Name`, which is the internal name.

You cannot delete existing mappings in the sample value. However, you can modify these mappings.

If you want to add mappings for fields other than the ones already present in the alias entry, then you can add them either to the existing values in the alias entry, or add them to the alias + entry.

The following is the default value of the alias + entry:

```
['__ENABLE__':'Status', 'FirstName':'First Name', 'email':'Email',
'JoiningDate':'Start Date]
```

The following is the format in which you must specify values for the alias and alias + entry:

['*CONN_ATTR1*': '*OIM_FIELD1*', '*CONN_ATTR2*': '*OIM_FIELD2*', . . . '*CONN_ATTRn*': '*OIM_FIELDn*']

In this format:

* *CONN_ATTR* is the connector attribute name.
* *OIM_FIELD* is the name of the field on the Oracle Identity Manager User form.

– For target resource configuration

In the target resource configuration section, you use the alias entry for one or all of the following purposes:

* To map connector attributes or target system attributes to fields of the process form. The mappings that you specify here are used to populate entries in the Recon Attribute map and Prov Attribute map lookup definitions for target resource reconciliation.
* To set an alias (a unique and shortened name) for the IT resource name specified in the itResourceName entry.
* To specify a short name for a lengthy process form field name.

  When the number of characters in a process form is more than 11, the metadata generator automatically truncates the process form name to 10 characters and then suffixes it with the digit 0. Subsequently, for every process form that results in the same name after truncating, the suffix is incremented by 1. The metadata generator prevents any two process forms from having the same name by using autonumbering. To gain control over the autogenerated form name and to have meaningful form names, you can use an alias to specify a shortened process form name.

  This is illustrated by the following example:

  Assume that the resource name is GENDB and contains child data that is represented as USER_ROLES in the schema.

  When you run the metadata generator, the process form is created and the form name is UD_GENDB_USER_ROLES. As the number of characters in this process form name is more than 11, the metadata generator automatically truncates it to UD_GENDB_U0. The truncated form name, UD_GENDB_U0, is not meaningful.

  To avoid encountering such issues or forms with autogenerated names, you can use the alias entry to specify short and meaningful process form names.

  The following is the default value of the alias entry in the target resource configuration section:

  ```
  [‘__UID__':'userId', '__NAME__':'userName']
  ```

  You cannot delete existing mappings in the default value as they are mandatory. However, you must modify the default value to match the values of the UidAttribute and NameAttribute qualifiers in the schema file. For example, in the schema file, if you have set the values of the UidAttribute and NameAttribute qualifiers to UID and primaryEmail respectively, then you must set the value of the alias entry to the following:

  ```
  ['__UID__':'UID', '__NAME__':primaryEmail]
  ```

If you want to add mappings for fields other than the ones already present in the alias entry (in other words, optional aliases), then you can add them either to the existing values in the alias entry, or add them to the alias + entry.

The following is the default value of the alias + entry in the target resource section:

```
['Generic Scim Target':'GST', 'comments':'Description', 'Family Name':'Last Name', 'Visibility':'Status']
```

The following is the format in which you must specify values for the alias and alias + entries:

['*CONN_ATTR1*': '*ALIAS_FIELD1*', '*CONN_ATTR2*': '*ALIAS_FIELD2*', . . . '*CONN_ATTRn*': '*ALIAS_FIELDn*']

In this format:

* *CONN_ATTR* is the connector attribute name.
* *ALIAS_FIELD* is the alias corresponding to the connector attribute or target system attribute.

- prepopulate

  This is an optional entry that is present only in the section for target resource configuration. Specify a value for this entry if you want Oracle Identity Manager to prepopulate connector's process form fields from Oracle Identity Manager User fields while provisioning a enterprise target system resource.

  The sample value of this entry is as follows:

  ```
  ['__NAME__':'User Login', 'FIRST_NAME':'First Name', 'LAST_NAME':'Last Name', '__PASSWORD__':'Password']
  ```

  This means that the groovy file is configured to prepopulate the following fields by default:

  – User Login
  – First Name
  – Last Name
  – Password

  You can add fields to or remove fields from the preceding list. The following is the format in which you must specify values for the prepopulate entry:

  ['*CONN_ATTR1* or *TARGET_ATTR1*': '*OIM_FIELD1*', '*CONN_ATTR2* or *TARGET_ATTR2*': '*OIM_FIELD2*', . . . '*CONN_ATTRn* or *TARGET_ATTRn*': '*OIM_FIELDn*']

  In this format:

  – *CONN_ATTR* is the connector attribute name.
  – *TARGET_ATTR* is the target system attribute name.
  – *OIM_FIELD* is the name of the field on the Oracle Identity Manager User form.

  See Working with Prepopulate Adapters in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager* for more information about attaching and removing prepopulate adapters.

2.2.3 Updating the Groovy File

Oracle Identity Manger provides an option to edit or configure an existing groovy file. Using a text editor, you can configure the groovy file as per requirements.

To configure the GenericScimConfiguration.groovy file:

1. Download the connector installation ZIP file from Oracle Technology Network.
2. Extract the contents of the connector installation ZIP to any directory on the computer hosting Oracle Identity Manager . This creates a directory named genericSCIM-*RELEASE_NUMBER.* See Files and Directories of the Generic SCIM Connector for information about all the files and directories in the connector installation ZIP.
3. In a text editor, open the GenericScimConfiguration.groovy file located in the genericSCIM-*RELEASE_NUMBER*/metadata-generator/resources directory.
4. Specify values for entries in one of the following predefined sections:
   - trusted - for configuring the connector for trusted source mode.
   - target - for configuring the connector for target resource mode.

   > **Note:**
   >
   > See Understanding Entries in the Predefined Sections of the Groovy File.

5. Save and close the GenericScimConfiguration.groovy file.

2.3 Generating the Generic SCIM Connector

You generate a connector package zip for your target system by running the metadata generator.

After configuring the GenericScimConfiguration.groovy file, you must run the metadata generator to generate the connector package based on your target system schema.

The metadata generator is the GenericSCIMGenerator.cmd or GenericSCIMGenerator.sh file that is located in the genericscim-*RELEASE_NUMBER*/metadata-generator/bin directory.

To run the metadata generator, in a command window, change to the genericscim-*RELEASE_NUMBER*/metadata-generator/bin directory (for example, genericscim-11.1.1.5.0/bin) and run one of the following commands depending on the operating system that you are using:

- **For Microsoft Windows**

  ```
  GenericSCIMGenerator.cmd CONFIG_FILE CONFIG_NAME
  ```

- **For UNIX**

  ```
  GenericSCIMGenerator.sh CONFIG_FILE CONFIG_NAME
  ```

In this command, replace:

- *CONFIG_FILE* with the absolute or relative path name of the GenericScimConfiguration.groovy file.
- *CONFIG_NAME* with the name of the configuration within the GenericScimConfiguration.groovy file, being used for the target system. The predefined configurations within this file are trusted and target. You can create additional custom configurations with different names depending on your requirements.

  Sample command:
  `GenericSCIMGenerator.cmd ..\resources\GenericScimConfiguration.groovy target`

  In this command, `target` denotes the name of the section in the `GenericScimConfiguration.groovy file` for which values have been specified. In other words, the connector is being configured for the target resource mode. If you encounter any errors while running the metadata generator, then you must fix it and then resume running the metadata generator.

2.3.1 Understanding the Generated Connector Package for the Generic SCIM Connector

The connector package is a ZIP file that is generated in the GenericSCIM-*RELEASE_NUMBER*/metadata-generator/ directory. For example, if you have specified GenSCIM as the value of the itResourceDefName entry in the GenericScimConfiguration.groovy file, then the connector package ZIP (GenSCIM.zip) file is generated in the GenericSCIM-11.1.1.5.0/metadata-generator/directory.

The directory structure of the connector package is as follows:

```
CONNECTOR_PACKAGE/
        configuration/
                 IT_RES_DEF-CI.xml
        resources/
                 genericscim-generator.properties
        xml/
                IT_RES_DEF-ConnectorConfig.xml
```

In this directory structure:

- *CONNECTOR_PACKAGE* is replaced with the name of the IT resource definition specified as the value of the itResourceDefName entry in the GenericScimConfiguration.groovy file.
- *IT_RES_DEF* is replaced with the name of the IT resource definition specified as the value of the itResourceDefName entry in the GenericScimConfiguration.groovy file.

The following behavior is observed after generation of the connector configuration XML file:

The length of a field (column) from the target system is not fetched into the process form. Therefore, except for the Unique ID and Password fields, the length of all other data fields (of the String data type) on the process form is always set to 255 characters. The length of the Unique ID and Password fields is set to 40 characters.

3

Installing and Configuring the Generic SCIM Connector

You must install the connector in Oracle Identity Manager. If necessary, you can also deploy the connector in a Connector Server.

The following topics provide details to install and configure the Generic SCIM connector:

- Preinstallation
- Installing the Generic SCIM Connector
- Postinstallation

3.1 Preinstallation

Preinstallation for the Generic SCIM connector involves custom authentication implementation and custom parsing implementation. For the SCIM connector, the preinstallation steps are performed before the metadata generation.

The preinstallation steps include the following optional procedures:

- Implementing Custom Authentication
- Implementing Custom Parsing

3.1.1 Implementing Custom Authentication

If your target system uses an authentication mechanism that is not supported by this connector, then you must implement the authentication that your target system uses and then attach it to the connector by using the plug-ins exposed by this connector. Implementing custom authentication involves creating a Java class, overriding the Map<String, String> getAuthHeaders(Map<String, Object> authParams) method that returns the authorization header in the form of a map, and updating the connector installation media to include the new Java class. All the target system configuration and authentication details that may be required for obtaining the authorization header are passed to the Map<String, String> getAuthHeaders(Map<String, Object> authParams) method through specific IT resource parameters. All the configuration properties exposed by this connector are accessible within this method as a part of "authParams".

To implement a custom authentication:

1. Create a Java class for implementing custom authentication. This class must implement the org.identityconnectors.scimcommon.auth.spi.AuthenticationPlugin interface.

   Note down the name of this Java class. You will provide the name of the Java class while configuring the IT resource for your target system which is described later in this guide.

2. Override the **Map<String, String> getAuthHeaders(Map<String, Object> authParams)** method in the custom Java class.

   This method must implement the custom authentication logic that returns the authorization header in the form of a map For example, `{ Authorization = Bearer XXXXXXXXXX }`. The authorization header contains the access token received from the target.

3. Package the Java class implementing the custom authentication into a JAR file.

4. Package the JAR file containing the custom authentication implementation with the connector bundle JAR as follows:

   > **Note:**
   >
   > Ensure to package all the JARs for any other custom implementations that you may have.

   a. Extract the contents of the org.identityconnectors.genericscim-1.0.1115.jar file into a temp directory. This file is located in the GenericSCIM-***RELEASE_NUMBER***\bundle directory.

   b. Copy the JAR file containing the custom authentication (from Step 3) to the lib directory.

   c. Regenerate the connector bundle (org.identityconnectors.genericscim-1.0.1115.jar) by running the following command:

   ```
   jar -cvfm org.identityconnectors.genericscim-1.0.1115.jar META-INF/MANIFEST.MF *
   ```

   > **Note:**
   >
   > While updating the connector bundle, ensure that META-INF\MANIFEST.MF file is unchanged.

5. Run the Oracle Identity Manager Delete JARs utility to delete any existing JARs in Oracle Identity Manager database before you upload the regenerated connector bundle. This utility is copied into the following location when you install Oracle Identity Manager:

   > **Note:**
   >
   > Before you use this utility, verify that the `WL_HOME` environment variable is set to the directory in which Oracle WebLogic Server is installed.

   - **For Microsoft Windows**:*OIM_HOME*/server/bin/DeleteJars.bat
   - **For UNIX**: *OIM_HOME*/server/bin/DeleteJars.sh

When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being deleted, and the location from which the JAR file is to be deleted. Specify 4 (ICF Bundle) as the value of the JAR type.

6. Run the Oracle Identity Manager Upload JARs utility to upload the regenerated connector bundle to Oracle Identity Manager database. This utility is copied into the following location when you install Oracle Identity Manager:

> **Note:**
>
> Before you use this utility, verify that the `WL_HOME` environment variable is set to the directory in which Oracle WebLogic Server is installed.

   - **For Microsoft Windows**:*OIM_HOME*/server/bin/UploadJars.bat
   - **For UNIX**: *OIM_HOME*/server/bin/UploadJars.sh

   When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being uploaded, and the location from which the JAR file is to be uploaded. Specify 4 (ICF Bundle) as the value of the JAR type.

7. Restart Oracle Identity Manager.

This completes the procedure for implementing a custom authentication.

3.1.2 Implementing Custom Parsing

By default, the connector supports only JSON parsing during reconciliation runs. If the reconciliation data from your target system is not in JSON format, then you must write a custom parser implementation for your data format.

To implement custom parsing:

1. Create a Java class for implementing the custom parser. This class must implement the org.identityconnectors.scimcommon.parser.spi.ParserPlugin interface.

   Note down the name of this Java class. You will provide the name of the Java class while configuring the IT resource for your target system which is described later in this guide.

2. Override the **String parseRequest(Map<String, Object> attrMap)** and **List<Map<String, Object>> parseResponse(String response, Map<String, String> parserConfigParams)** methods in the custom Java class.

   The String parseRequest(Map<String, Object> attrMap) method implements the logic for parsing an attribute and generates a string request payload.

   The List<Map<String, Object>> parseResponse(String response, Map<String, String> parserConfigParams) method implements the logic for parsing the string response received from the target in this class.

3. Package the Java class implementing the custom parser into a JAR file.

4. Package the JAR file containing the custom parser implementation with the connector bundle JAR as follows:

   > **Note:**
   >
   > Ensure to package all the JARs for any other custom implementations that you may have.

   a. Extract the contents of the org.identityconnectors.genericscim-1.0.1115.jar file into a temp directory. This file is located in the GenericSCIM-***RELEASE_NUMBER***\bundle directory.

   b. Copy the JAR file containing the custom authentication (from Step 3) to the lib directory.

   c. Regenerate the connector bundle (org.identityconnectors.genericscim-1.0.1115.jar) by running the following command:

   ```
   jar -cvfm org.identityconnectors.genericscim-1.0.1115.jar META-
   INF/MANIFEST.MF *
   ```

   > **Note:**
   >
   > While updating the connector bundle, ensure that META-INF\MANIFEST.MF file is unchanged.

5. Run the Oracle Identity Manager Delete JARs utility to delete any existing JARs in Oracle Identity Manager database before you upload the regenerated connector bundle. This utility is copied into the following location when you install Oracle Identity Manager:

   > **Note:**
   >
   > Before you use this utility, verify that the `WL_HOME` environment variable is set to the directory in which Oracle WebLogic Server is installed.

   - **For Microsoft Windows**:*OIM_HOME*/server/bin/DeleteJars.bat
   - **For UNIX**: *OIM_HOME*/server/bin/DeleteJars.sh

   When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being deleted, and the location from which the JAR file is to be deleted. Specify 4 (ICF Bundle) as the value of the JAR type.

6. Run the Oracle Identity Manager Upload JARs utility to upload the regenerated connector bundle to Oracle Identity Manager database. This utility is copied into the following location when you install Oracle Identity Manager:

> **Note:**
>
> Before you use this utility, verify that the `WL_HOME` environment variable is set to the directory in which Oracle WebLogic Server is installed.

- **For Microsoft Windows**:*OIM_HOME*/server/bin/UploadJars.bat
- **For UNIX**: *OIM_HOME*/server/bin/UploadJars.sh

When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being uploaded, and the location from which the JAR file is to be uploaded. Specify 4 (ICF Bundle) as the value of the JAR type.

7. Restart Oracle Identity Manager.

This completes the procedure for implementing custom parsers.

3.2 Installing the Generic SCIM Connector

You must install the connector in Oracle Identity Manager. If necessary, you can also deploy the connector in a Connector Server.

The following topics provide details on installing the connector:

- Understanding Installation of the Generic SCIM Connector
- Running the Connector Installer
- Configuring the IT Resource for the Target System

3.2.1 Understanding Installation of the Generic SCIM Connector

The procedure to understand installation of the Generic SCIM Connector is divided across two stages namely summary of steps to install the connector and about installing the Generic SCIM connector locally and remotely.

- Summary of Steps to Install the Connector
- About Installing the Generic SCIM Connector Locally and Remotely

3.2.1.1 Summary of Steps to Install the Connector

Installing this connector requires you to install the connector bundle that is included in the installation media and then install the connector package (specific to your target system) that you had generated while performing the procedure described in Generating the Generic SCIM Connector section.

The following is a summary of steps to install the Generic SCIM connector:

1. Run the connector installer to install the connector bundle included in the installation media. This procedure is described later in this chapter.
2. Run the connector installer to install the connector package (specific to your target system) that you had generated while performing the procedure described in

Generating the Generic SCIM Connector. The procedure to install the connector package is described later in this guide.

3. Configure the IT resource. See Configuring the IT Resource for the Target System.

3.2.1.2 About Installing the Generic SCIM Connector Locally and Remotely

You can run the connector code either locally in Oracle Identity Manager or remotely in a Connector Server.

Depending on where you want to run the connector code (bundle), the connector provides the following installation options:

- Run the connector code locally in Oracle Identity Manager.

  In this scenario, you deploy the connector in Oracle Identity Manager. Deploying the connector in Oracle Identity Manager involves performing the procedures described in Running the Connector Installer and Configuring the IT Resource for the Target System.

- Run the connector code remotely in a Connector Server.

  In this scenario, you deploy the connector in Oracle Identity Manager, and then, deploy the connector bundle in a Connector Server. See Using an Identity Connector Server in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager* for information about installing, configuring, and running the Connector Server, and then installing the connector in a Connector Server.

3.2.2 Running the Connector Installer

When you run the Connector Installer, it automatically copies the connector files to directories in Oracle Identity Manager, imports connector XML files, and compiles adapters used for provisioning.

As discussed in one of the earlier sections, you must first install the connector bundle that is included in the installation media and then install the connector bundle that is a part of the connector package that you generated. The procedure to install both connector bundles is the same except for the following differences:

- Before running the connector installer to install the connector bundle from the installation media, you must copy the contents of the connector installation media to the *OIM_HOME*/server/ConnectorDefaultDirectory directory.
- Before running the connector installer to install the generated connector, you must copy the unzipped connector package (generated in Generating the Generic SCIM Connector) to the *OIM_HOME*/server/ConnectorDefaultDirectory directory.

You must install the connector in Oracle Identity Manager by using the Connector Installer. To do so:

1. If you are installing the connector included in the installation media, then copy the contents of the connector installation media to the following directory:

   OIM_HOME/server/ConnectorDefaultDirectory

2. If you are installing the connector from the generated connector package, then copy the unzipped connector package (generated in Generating the Generic SCIM Connector) to the following directory:

OIM_HOME/server/ConnectorDefaultDirectory

3. Log in to Oracle Identity System Administration.
4. In the left pane, under Provisioning Configuration, click **Manage Connector.**
5. In the Manage Connector page, click **Install.**
6. From the Connector List, select one of the following connectors:
   - If are installing the connector included in the connector installation media, then select **Generic SCIM Connector-*RELEASE_NUMBER.***
   - If you are installing the generated connector, then select the name of the connector package (generated by running the metadata generator).

   This list displays the names and release numbers of connectors whose installation files you copy into the default connector installation directory in Step 1.

   If you have copied the installation files into a different directory, then:

   a. In the **Alternative Directory** field, enter the full path and name of that directory.
   b. To repopulate the list of connectors in the Connector List list, click **Refresh.**
   c. From the Connector List list, select the relevant connector name depending on whether you are installing the connector included in the connector installation media or the generated connector.
7. Click **Load**.
8. To start the installation process, click **Continue**.

   The following tasks are performed in sequence:

   a. Configuration of connector libraries
   b. Import of the connector XML files (Using Deployment Manager).
   c. Compilation of Adapter Definitions

   On successful completion of a task, a check mark is displayed for the task. If a task fails, then an X mark and a message stating the reason for failure are displayed. Depending on the reason for the failure, make the required correction and then perform one of the following steps:

   - Retry the installation by clicking **Retry.**
   - Cancel the installation and begin again from Step 1.

   If all three tasks of the connector installation process are successful, then a message indicating successful installation is displayed.
9. Click **Exit** to close the installation page.

   When you run the Connector Installer, it processes the script in the GenericSCIM-CI.xml file located in the configuration directory. This file is listed in Files and Directories of the Generic SCIM Connector.

3.2.3 Configuring the IT Resource for the Target System

The IT resource for your target system is created after you install the connector. An IT resource is composed of parameters that store connection and other generic information about a target system. Oracle Identity Manager uses this information

to connect to a specific installation or instance of your target system and perform reconciliation and provisioning operations.

The list of IT resource parameters for the Generic SCIM connector can be grouped into the following categories:

- Connection-related parameters
- Authentication parameters
- Parser parameters
- Additional configuration parameters

> **Note:**
>
> The list of parameters that are displayed on the IT resource page depends on the properties that you added to the Config entry of the GenericSCIMConfiguration.groovy file. At any point in time, you can update the list of IT resource parameters by modifying the IT Resource Type definition using Oracle Identity Manager Design Console. There is no need to re-create and install the connector when you update the IT Resource Type definition.

The following topics related to IT resource configuration are discussed in this section:

- About IT Resource Parameter Categories
- IT Resource Parameters
- Specifying Values for the IT Resource Parameters

3.2.3.1 About IT Resource Parameter Categories

An IT resource is composed of parameters that store connection and other generic information about a target system. Oracle Identity Manager uses this information to connect to a specific installation or instance of your target system.

The list of IT resource parameters for this connector can be grouped into the following categories:

- Connection-related parameters
- Authentication parameters
- Parser parameters
- Configuration parameters

Connection Parameters

Connection parameters are used by the connector to establish a connection between Oracle Identity Manager and your target system for exchange of identity information.

Authentication Parameters

Authentication parameters are used by the target system to authenticate the application. The IT resource parameters for authentication vary depending on the value of the grantType parameter. The grantType parameter holds the type of

authentication used by your target system. By default, the connector supports the following types of authentication:

- Basic authentication
- OAuth2.0 JWT
- OAuth2.0 Client Credentials
- OAuth2.0 Resource Owner password

Apart from the authentication types listed, if you target system uses any other authentication type, then you must write your own implementation which requires development effort. The following are the possible values for this parameter:

- For HTTP Basic Authentication: `basic`
- For OAuth 2.0 JWT: `jwt`
- For OAuth 2.0 Client Credentials: `client_credentials`
- For OAuth 2.0 Resource Owner Password: `password`
- For custom authentication implementation: `custom`

Parser Parameters

By default, the connector supports only JSON parsing during reconciliation runs. If the reconciliation data from your target system is not in JSON format, then you must write a custom parser implementation for your data format. If the data from your target system is in JSON format, then the connector uses JSON parsing and you must provide a value for the jsonResourcesTag parameter. The jsonResourcesTag parameter must contain the json tag value that is used during reconciliation for parsing multiple entries in a single response payload. If you are using a custom parser implementation, then you must provide values for the parameters listed in table Table 3-7.

Additional Configuration Parameters

All additional configuration parameters are target system specific.

3.2.3.2 IT Resource Parameters

The IT resource for the target system contains connection information about the target system. Oracle Identity Manager uses this information during provisioning and reconciliation.

Connection Parameters

Table 3–1 lists the connection-related IT resource parameters.

Table 3-1 Connection IT Resource Parameters

Parameter	Description
schemaFile	Enter the name and relative path of the schema file that you want to use. See Defining the Schema for information about the schema file that you created.

Table 3-1 (Cont.) Connection IT Resource Parameters

Parameter	Description
username	User name or ID of the target system user account that Oracle Identity Manager uses to connect to the target system.
host	Host name or IP address of the computer hosting the target system. Sample value: `www.example.com`
port	Port number at which the target system is listening. Sample value: `80`
proxyHost	Proxy host is the name of the proxy host used to connect to an external target system. Sample value: `www.example.com`
proxyPort	Proxy port number Sample value: `80`
proxyUser	Proxy user ID of the target system user account that Oracle Identity Manager uses to connect to the target system.
proxyPassword	Password of the proxy user ID of the target system user account that Oracle Identity Manager uses to connect to the target system.
connectionTimeOut	An integer value that specifies the number of milliseconds after which an attempt to establish the connection between the target system and Oracle Identity Manager times out. Sample value: `100000`
socketTimeOut	An integer value that specifies the number of milliseconds after which the wait for a response from the target system times out. Sample value: `100000`
baseURI	Base URI refers to the base relative URL of the SCIM target system. For example if the URL is `http://host:port/hcmCoreSetupApi/scim/User`, then baseURI is `/hcmCoreSetupApi/scim`.
nameAttributes	Enter the name attribute for all object classes that are handled by the connector. This value specifies the mapping between the `_NAME_` connector attribute and the corresponding target system attribute for each object class that the connector handles. Format: *OBJ_CLASS.ATTR_NAME* **Note**: All values in this parameter must be comma separated.

Table 3-1 (Cont.) Connection IT Resource Parameters

Parameter	Description
uidAttributes	Enter the mapping between the _UID_ (GUID) connector attribute and target attribute for each object class that the connector handles. Format: *OBJ_CLASS.ATTR_NAME* **Note**: All values in this parameter must be comma separated.
statusAttributes	Enter the mapping between the _ENABLE_ connector attribute the target attribute that holds the status for each object class this connector handles. Format: *`OBJ_CLASS.ATTR_NAME`* **Note**: All values in this parameter must be comma separated.
grantType	Specifies the authorization grant used by your target system. The following are the supported grant types and the possible values for this property: – HTTP Basic Authentication — `basic` —OAuth2.0 JWT — `jwt` – OAuth 2.0 Client Credentials — `client_credentials` – OAuth 2.0 Resource Owner Password — `password` – If you have written your own custom implementation for authentication, then the value is `custom`
contentType	This parameter holds the content type expected by the target system in the header. The content type can be `application/scim+json.`
acceptType	This parameter holds the accept type expected by the target system in the header. The accept type can be `application/scim+json`.
scimVersion	This entry holds the SCIM version of the target system. The valid range for this attribute is `1` to `19.`

Table 3-1 (Cont.) Connection IT Resource Parameters

Parameter	Description
jsonResourcesTag	Enter the JSON tag value that is used for parsing a response payload. The value mentioned in this parameter will be considered as an unwanted outer tag while parsing response. You can skip entering a value for this parameter if there is no unwanted outer tag in your response payload. Enter a value for the parameter in the following format: *OBJ_CLASS=OUTER_ATTR_NAME* In this format, `OBJ_CLASS` is the name of the object class for which a response payload is being parsed. OUTER_ATTR_NAME is the name of the outer tag in the response payload. For example, consider the following JSON value for a User object: `"Resources": "{ "user1":"{value1}", "user2":"{value2}" }` From this example, the value of the jsonResourcesTag parameter is `__ACCOUNT__=Resources`. **Note**: You must enter a value for this parameter only if the data from your target system is in JSON format. For more than one JSON tag, the values must be comma separated.
attrToOClassMapping	This entry is used to map an attribute of one object class with another object class. For example, if the groups attribute of the `__ACCOUNT__` object class must be mapped to the `__GROUP__` object class, then enter `__ACCOUNT__.groups=__GROUP__`. Sample value: *__ACCOUNT__.groups=__GROUP__*

Authentication Parameters

As discussed in one of the earlier sections, IT resource parameters for authentication vary depending on the value that you specify for the grantType parameter.

Table 3-2 lists the set of IT resource parameters for which you must enter values when the grantType parameter is set to basic.

Table 3-2 HTTP Basic Authentication IT Resource Parameters

Parameter	Description
username	User name or User ID of the account that Oracle Identity Manager must use to connect to and access the target system during reconciliation and provisioning operations. Sample value: `johnsmith`
password	Password of the account that Oracle Identity Manager must use to connect to and access the target system during reconciliation and provisioning operations. Sample value: `password`

Table 3-3 lists the set of IT resource parameters for which you must enter values when the grantType parameter is set to `jwt`.

Table 3-3 OAuth 2.0 JWT IT Resource Parameters

Parameter	Description
aud	Enter the intended audience of the JWT. The value can either be a URI or token endpoint URL of the authorization server. Sample value: `https://www.example.com/oauth2/v3/token`
iss	Enter a value that uniquely identifies the entity that issued the JWT. Sample value: `527901474-ugnvd5uh21p598cf9h6cd@developer.example.com`
scope	Enter the scope of the access token being issued. Sample value: `https://www.example.com/auth/adm.direct.group`, `https://www.example.com/auth/adm.direct.user`
sub	Enter a value that identifies the principal to which the JWT is being issued. Sample value: `admin@example.com`
privateKeyLocation	Enter the absolute path to the private key used to sign the access token. Sample value: `D:\\SCIM Connector – 1595b926.p12`
privateKeySecret	Enter the secret key for the private key that is being used to sign the access token.
tokenLifespan	Enter the life span of the access token in milliseconds. Sample value:`3600`

Table 3-3 (Cont.) OAuth 2.0 JWT IT Resource Parameters

Parameter	Description
signatureAlgorithm	Enter the algorithm used for signing the access token. Sample value: `RS265`
privateKeyFormat	Enter the format of the private key used to sign the access token. Sample Value:`PKCS12`

Table 3-4 lists the set of IT resource parameters for which you must enter values when the grantType parameter is set to `client_credentials`.

Table 3-4 OAuth2.0 Client Credentials IT Resource Parameters

Parameter	Description
clientId	Enter the client identifier (a unique string) issued by the authorization server to the client during the registration process. Sample value: `XDWTh0r2eWuULCDVt`
clientSecret	Enter the value used to authenticate the identity of your client application. Sample value: `clZsdZisTOoYN5NITirarIDepDkiJTGHdzNFT0m`
authenticationServerURL	Enter the URL of the authorization server that authenticates the client (by validating the client ID and client secret), and if valid, issues an access token. Sample value: `https://api.example.com/oauth2/token`

Table 3-5 lists the set of IT resource parameters for which you must enter values when the grantType parameter is set to `password`.

Table 3-5 OAuth2.0 Resource Owner Password IT Resource Parameters

Parameter	Description
username	Enter the user name or user ID of the resource owner. Sample value: `johnsmith`
password	Enter the password of the resource owner. Sample value: `password`
clientId	Enter the client identifier issued to the client during the registration process. Sample value: `XDWTh0r2eWuULCDVt`

Table 3-5 (Cont.) OAuth2.0 Resource Owner Password IT Resource Parameters

Parameter	Description
clientSecret	Enter the client secret used to authenticate the identity of the client application. Sample value: `clZsdZisTOoYN5NITirarIDepDkiJTGHdzNFT0m`
authenticationServerUrl	Enter the URL of the authorization server (token endpoint) that authenticates the client (by validating client ID and client secret) and the resource owner credentials, if valid, issues an access token. Sample value: `https://api.example.com/oauth2/token`

Table 3-6 lists the set of IT resource parameters that are displayed when the grantType parameter is set to `custom`.

Table 3-6 Custom Implementation IT Resource Parameters

Parameter	Description
customAuthClassName	Enter the name of the class implementing the custom authentication logic that you created while performing the procedure described in Implementing Custom Authentication.
customAuthConfigParams	Enter any configuration parameters that you may use in the custom authentication class *PARAM_NAME1*=VAL1,*PARAM_NAME2*=VAL2, . . . *PARAM_NAMEn*=VALn

Parser Parameters

Table 3-7 lists the set of IT resource parameters when the data from your target system is reconciled in a custom parser implementation, other than JSON format.

Table 3-7 Custom Parser IT Resource Parameters

Parameter	Description
customParserClassName	Enter the name of the class implementing the custom parser logic that you created while performing the procedure described in Implementing Custom Parsing.
customParserConfigParams	Enter any configuration parameters that you may use in the custom parser class. You must enter a value for this parameter in the following format: *PARAM_NAME1*=VAL1,*PARAM_NAME2*=VAL2, . . . *PARAM_NAMEn*=VALn

Additional Configuration Parameters

All additional configuration parameters are target system specific. Table 3-8 lists the IT resource parameters related to target system configuration. The supported operation types for all the parameters listed in this table are CREATEOP, DELETEOP, SEARCHOP, and UPDATEOP.

Table 3-8 Configuration IT Resource Parameters

Parameter	Description
sslEnabled	Specifies whether SSL communication is enabled between Oracle Identity Manager and your target system. Enter yes if SSL is configured. Otherwise, enter no.

Table 3-8 (Cont.) Configuration IT Resource Parameters

Parameter	Description
relURLs	Enter the relative URLs for all operations of each object class. Enter a value for this parameter in one of the following formats: • For attributes: *OBJ_CLASS.OP=REL_URL* • For special attributes: *OBJ_CLASS.ATTR_NAME.OP=REL_URL* • For attributes that have the same relative URL for multiple operations: *OBJ_CLASS=REL_URL* or *OBJ_CLASS.ATTR_NAME=REL_URL* • If you have to pass the unique ID of the user as part of endpoint URL, use $(__UID__)$. • If you have to pass any attribute other than the unique ID of the user, then represent it in one of the following formats: – For a single-valued attribute: $(firstname)$ – For an embedded object: $(*OBJ_CLASS*.*ATTR_NAME*)$ For example, $(__GROUP__.id)$. **Note:** All values in this parameter must be comma separated. Sample value: "__ACCOUNT__.CREATEOP=/admin/directory/v1/users","__ACCOUNT__.SEARCHOP=/admin/directory/v1/users/$(FilterSuffix)$","__ACCOUNT__=/admin/directory/v1/users/$(__UID__)$","__GROUP__.CREATEOP=/admin/directory/v1/groups","__GROUP__=/admin/directory/v1/groups/$(__UID__)$","__GROUP__.SEARCHOP=/admin/directory/v1/groups/$(FilterSuffix)$","__ACCOUNT__.alias=/admin/directory/v1/users/$(__UID__)$/aliases","__ACCOUNT__.alias.DELETEOP=/admin/directory/v1/users/$(__UID__)$/aliases/$(alias)$","__GROUP__.alias.DELETEOP=/admin/directory/v1/groups/$(__UID__)$/aliases/$(alias)$","__ACCOUNT__.__GROUP__=/admin/directory/v1/groups/$(__GROUP__.id)$/members","__ACCOUNT__.__GROUP__.DELETEOP=/admin/directory/v1/groups/$(__GROUP__.id)$/members/$(__UID__)$","__ACCOUNT__.__GROUP__.S

Table 3-8 (Cont.) Configuration IT Resource Parameters

Parameter	Description
	`EARCHOP=/admin/directory/v1/groups?userKey=$(__UID__)$"`
customHeaders	Enter any custom or additional header values that must be sent to the target system. Format: `"HEADER_NAME1=VALUE1","HEADER_NAME2=VALUE2", . . . "HEADER_NAMEn=VALUEn"` **Note**: If you are using a SCIM target as Oracle Identity Governance 12*c* (12.2.1.4.0), then enter an additional header for post request. For example: `"X-REQUESTED-BY=test"`
customAuthHeaders	Enter any additional header values that must be sent to the target system only during authentication. If you are entering a value for this parameter as you have set the grantType parameter to other, then enter the access token and refresh token values that must be passed through an HTTP authorization header.
customPayload	Enter a comma-separated list of request payload formats for target system attributes that do not adhere to the standard JSON format. Format: *OBJ_CLASS.ATTRNAME.OP=PAYLOAD_FORMAT* **Note:** If you must pass the unique ID of the user as part of a custom payload, then represent it as `$(__UID__)$`. If you must pass the value of any other attribute, then represent it as `$(`*`ATTRIBUTE_NAME`*`)$` . Sample value: `"__ACCOUNT__.__GROUP__.UPDATEOP={ \"user\": { \"id\": \"$(__UID__)$\"}, \"group\": { \"id\": \"$(id)$\" } }"`
dateAttributes	Specifies a list of date attributes available on the target system. Sample value: `"Users=meta.lastModified","Groups=meta.lastModified"`
passwordAttribute	Specifies the mapping between `__PASSWORD__` (password) of the connector with the target system attribute for each object class. Format: `objectClass=attributeName` **Note**: All values in this parameter must be comma separated.
dateFormat	Specifies date format of the date attributes available on the target system. Sample value: `MMM d, yyyy h:mm:ss a z`

Table 3-8 (Cont.) Configuration IT Resource Parameters

Parameter	Description
lookupObjectClasses	Specifies a list of object class that is used for scheduled tasks. This list of object class is not available by default on the target system.
httpOperationTypes	Specifies the type of HTTP Operation that needs to be performed for a particular operation on the attribute of an object class. Sample Value : *"__ACCOUNT__.password.UpdateOp=PUT"* **Note:** The connector supports only the PATCH method to perform Modify or Update operations from Oracle Identity Manager to a SCIM-based target system.
defaultBatchSize	This holds the default page/batch size for the GET operations. Default value: 500
reconSortByAttrs	Specifies an attribute name and the value. Based on this value, the sorting of the GET operation is performed by the target system. Sample value: *Users=id","Groups=id*

3.2.3.3 Specifying Values for the IT Resource Parameters

The IT resource for the target system contains connection information about the target system. Oracle Identity Manager uses this information during provisioning and reconciliation.

When you run the metadata generator, the IT resource corresponding to this connector is automatically created in Oracle Identity Manager. You must specify values for the parameters of this IT resource as follows:

1. Log in to Oracle Identity System Administration.
2. In the left pane, under Configuration, click **IT Resource.**
3. In the IT Resource Name field on the Manage IT Resource page, enter the name of the IT resource, and then click **Search.** The name of the IT resource is the value of the itResourceName property in the GenericScimConfiguration.groovy file.
4. Click the edit icon for the IT resource.
5. From the list at the top of the page, select **Details and Parameters**.
6. Specify values for the parameters of the IT resource. See the Configuring the IT Resource for the Target System section for information about IT resource parameters.
7. To save the values, click **Update.**

3.3 Postinstallation

Postinstallation for the connector involves configuring Oracle Identity Manager, enabling logging to track information about all connector events, and configuring SSL. It also involves performing some optional configurations such as localizing the user interface.

This topic discusses the following postinstallation procedures:

- Configuring Oracle Identity Manager
- Localizing Field Labels in UI Forms
- Clearing Content Related to Connector Resource Bundles from the Server Cache
- Managing Logging for the Generic SCIM Connector
- Configuring SSL for the Generic SCIM Connector

3.3.1 Configuring Oracle Identity Manager

You must create a UI form and an application instance for the resource against which you want to perform reconciliation and provisioning operations. In addition, you must run entitlement and catalog synchronization jobs.

> **Note:**
>
> Perform the procedures described in this section *only* if you are using the connector in the target resource configuration mode.

You must create a UI form and an application instance for the resource against which you want to perform reconciliation and provisioning operations. In addition, you must run entitlement and catalog synchronization jobs. These procedures are described in the following sections:

- Creating and Activating a Sandbox
- Creating a New UI Form
- Associating the Form with the Application Instance
- Publishing a Sandbox
- Harvesting Entitlements and Sync Catalog

3.3.1.1 Creating and Activating a Sandbox

You must create and activate a sandbox to begin using the customization and form management features. You can then publish the sandbox to make the customizations available to other users.

See Creating a Sandbox and Activating and Deactivating a Sandbox in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager.*

3.3.1.2 Creating a New UI Form

See Creating Forms by Using the Form Designer in *Oracle Fusion Middleware Administering Oracle Identity Manager* for instructions on creating a new UI form. While creating the UI form, ensure that you select the resource object corresponding to the Generic SCIM connector that you want to associate the form with. In addition, select the **Generate Entitlement Forms** check box.

3.3.1.3 Associating the Form with the Application Instance

By default, an application instance is automatically created after you install the connector. The name of this application instance is the one that is specified as the value of the applicationInstanceName entry in the GenericScimConfiguration.groovy file. If you did not specify a value for the applicationInstanceName entry, then the application instance name will be the same as the value of the ITResourceDefName entry.

You must associate this application instance with the form created in Creating a New UI Form.

See Managing Application Instances in *Oracle Fusion Middleware Administering Oracle Identity Manager* for instructions on modifying an application instance to associate it with a form.

After updating the application instance, you must publish it to an organization to make the application instance available for requesting and subsequent provisioning to users. However, as a best practice, perform the following procedure before publishing the application instance:

1. In the System Administration, deactivate the sandbox.
2. Log out of the System Administration.
3. Log in to the Self Service and activate the sandbox that you deactivated in Step 1.
4. In the Catalog, check for the Application Instance UI (form fields) and ensure that it appears correctly.
5. Publish the application instance only if everything appears correctly. Otherwise, fix the issues and then publish the application instance. See Publishing an Application Instance to Organizations in *Oracle Fusion Middleware Administering Oracle Identity Manager*.

3.3.1.4 Publishing a Sandbox

Before you publish a sandbox, perform the following procedure as a best practice to validate all sandbox changes made till this stage as it is hard to revert changes once a sandbox is published:

1. In the System Administration, deactivate the sandbox.
2. Log out of the System Administration.
3. Log in to the Self Service using the xelsysadm user credentials and then activate the sandbox that you deactivated in Step 1.
4. In the Catalog, ensure that the Generic SCIM application instance form appears with correct fields.

5. Publish the sandbox. See Publishing a Sandbox in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager*.

3.3.1.5 Harvesting Entitlements and Sync Catalog

To harvest entitlements and sync catalog:

1. Run the scheduled jobs for lookup field synchronization discussed in Scheduled Job for Lookup Field Synchronization.
2. Run the Entitlement List scheduled job to populate Entitlement Assignment schema from child process form table.
3. Run the Catalog Synchronization Job scheduled job.

> **See Also:**
>
> Predefined Scheduled Tasks in *Oracle Fusion Middleware Administering Oracle Identity Manager* .

3.3.2 Localizing Field Labels in UI Forms

You can localize UI form field labels by using the resource bundle corresponding to the language you want to use. Resource bundles are available in the connector installation media.

To localize field label that is added to the UI forms:

1. Create a properties file (for example, GS_ja.properties) containing localized versions for the column names in your target system (to be displayed as text strings for GUI elements and messages in the Oracle Identity Self Service).
2. Log in to Oracle Enterprise Manager.
3. In the left pane, expand **Application Deployments** and then select **oracle.iam.console.identity.sysadmin.ear.**
4. In the right pane, from the Application Deployment list, select **MDS Configuration.**
5. On the MDS Configuration page, click **Export** and save the archive to the local computer.
6. Extract the contents of the archive, and open the following file in a text editor:

   SAVED_LOCATION\xliffBundles\oracle\iam\ui\runtime\BizEditorBundle_en.xlf
7. Edit the BizEditorBundle.xlf file in the following manner:
   a. Search for the following text:

      ```
      <file source-language="en"
      original="/xliffBundles/oracle/iam/ui/runtime/
      BizEditorBundle.xlf"
      datatype="x-oracle-adf">
      ```

   b.

Replace with the following text:

```
<file source-language="en" target-language="LANG_CODE"
original="/xliffBundles/oracle/iam/ui/runtime/
BizEditorBundle.xlf"
datatype="x-oracle-adf">
```

In this text, replace *LANG_CODE* with the code of the language that you want to localize the form field labels. The following is a sample value for localizing the form field labels in Japanese:

```
<file source-language="en" target-language="ja"
original="/xliffBundles/oracle/iam/ui/runtime/
BizEditorBundle.xlf"
datatype="x-oracle-adf">
```

c. Search for the application instance code. This procedure shows a sample edit for Generic SCIM application instance. The original code is:

```
<trans-unit
id="$
{adfBundle['oracle.adf.businesseditor.model.util.BaseRuntimeResou
rceBundle']
['persdef.sessiondef.oracle.iam.ui.runtime.form.model.user.entity
.userEO.UD_ACMEGSAP_APP_DFLT_HOME__c_description']}">
<source>APP_DFLT_HOME</source>
<target/>
</trans-unit>
<trans-unit
id="sessiondef.oracle.iam.ui.runtime.form.model.ACMEFORM.entity.A
CMEFORMEO.UD_ACMEGSAP_APP_DFLT_HOME__c_LABEL">
<source>APP_DFLT_HOME</source>
<target/>
</trans-unit>
```

d. Open the properties file created in Step 1 and get the value of the attribute, for example, global.udf.D_ACMEGSAP_APP_DFLT_HOME=\u4567d.

e. Replace the original code shown in Step c with the following:

```
<trans-unit id="$
{adfBundle['oracle.adf.businesseditor.model.util.BaseRuntimeResou
rceBundle']
['persdef.sessiondef.oracle.iam.ui.runtime.form.model.user.entity
.userEO.UD_ACMEGSAP_APP_DFLT_HOME__c_description']}">
<source>APP_DFLT_HOME</source>
<target>\u4567d</target>
</trans-unit>
<trans-unit
id="sessiondef.oracle.iam.ui.runtime.form.model.ACMEFORM.entity.A
CMEFORMEO.UD_ACMEGSAP_APP_DFLT_HOME__c_LABEL">
<source>APP_DFLT_HOME</source>
<target>\u4567d</target>
</trans-unit>
```

f. Repeat Steps 7.a through 7.d for all attributes of the process form.

g. Save the file as BizEditorBundle_*LANG_CODE*.xlf. In this file name, replace *LANG_CODE* with the code of the language to which you are localizing.

   Sample file name: BizEditorBundle_ja.xlf.

h. Repackage the ZIP file and import it into MDS.

> **Note:**
>
> See Deploying and Undeploying Customizations in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager* for more information about exporting and importing metadata files.

i. Log out of and log in to Oracle Identity Manager.

3.3.3 Clearing Content Related to Connector Resource Bundles from the Server Cache

When you deploy the connector, the resource bundles are copied from the resources directory on the installation media into the Oracle Identity Manager database. Whenever you add a new resource bundle to the connectorResources directory or make a change in an existing resource bundle, you must clear content related to connector resource bundles from the server cache

To clear content related to connector resource bundles from the server cache you can either restart Oracle Identity Manager or run the PurgeCache utility. The following is the procedure to clear the server cache by running the PurgeCache utility:

1. In a command window, switch to the *OIM_HOME*/server/bin directory.
2. Enter one of the following commands:
   - **On Microsoft Windows**: `PurgeCache.bat All`
   - **On UNIX**: `PurgeCache.sh All`

   When prompted, enter the user name and password of an account belonging to the SYSTEM ADMINISTRATORS group. In addition, you are prompted to enter the service URL in the following format:

   `t3://OIM_HOST_NAME:OIM_PORT_NUMBER`

   In this format:

   - Replace *OIM_HOST_NAME* with the host name or IP address of the Oracle Identity Manager host computer.
   - Replace *OIM_PORT_NUMBER* with the port on which Oracle Identity Manager is listening.

   You can use the PurgeCache utility to purge the cache for any content category.

3.3.4 Managing Logging for the Generic SCIM Connector

Oracle Identity Manager uses Oracle Diagnostic Logging (ODL) logging service for recording all types of events pertaining to the connector.

The following topics provide detailed information about logging:

- Understanding Log Levels
- Enabling Logging

3.3.4.1 Understanding Log Levels

When you enable logging, Oracle Identity Manager automatically stores in a log file information about events that occur during the course of provisioning and reconciliation operations.

Oracle Identity Manager uses Oracle Java Diagnostic Logging (OJDL) for logging. OJDL is based on java.util.logger. To specify the type of event for which you want logging to take place, you can set the logs to one of the following available levels:

- SEVERE.intValue()+100

  This level enables logging of information about fatal errors.

- SEVERE

  This level enables logging of information about errors that might allow Oracle Identity Manager to continue running.

- WARNING

  This level enables logging of information about potentially harmful situations.

- INFO

  This level enables logging of messages that highlight the progress of the application.

- CONFIG

  This level enables logging of information about fine-grained events that are useful for debugging.

- FINE, FINER, FINEST

  These levels enable logging of information about fine-grained events, where FINEST logs information about all events.

These log levels are mapped to ODL message type and level combinations as shown in Table 3-9.

Table 3-9 Log Levels and ODL Message Type:Level Combinations

Log Level	ODL Message Type:Level
SEVERE.intValue()+100	INCIDENT_ERROR:1
SEVERE	ERROR:1
WARNING	WARNING:1
INFO	NOTIFICATION:1
CONFIG	NOTIFICATION:16
FINE	TRACE:1
FINER	TRACE:16

Table 3-9 (Cont.) Log Levels and ODL Message Type:Level Combinations

Log Level	ODL Message Type:Level
FINEST	TRACE:32

The configuration file for OJDL is logging.xml, which is located at the following path:

DOMAIN_HOME/config/fmwconfig/servers/*OIM_SERVER*/logging.xml

Here, *DOMAIN_HOME* and *OIM_SERVER* are the domain name and server name specified during the installation of Oracle Identity Manager.

3.3.4.2 Enabling Logging

To enable logging in Oracle WebLogic Server:

1. Edit the logging.xml file as follows:
   a. Add the following blocks in the file:

```
<log_handler name='genericscim-handler' level='[LOG_LEVEL]'
class='oracle.core.ojdl.logging.ODLHandlerFactory'>
<property name='logreader:' value='off'/>
     <property name='path' value='[FILE_NAME]'/>
     <property name='format' value='ODL-Text'/>
     <property name='useThreadName' value='true'/>
     <property name='locale' value='en'/>
     <property name='maxFileSize' value='5242880'/>
     <property name='maxLogSize' value='52428800'/>
     <property name='encoding' value='UTF-8'/>
   </log_handler>

<logger name="ORG.IDENTITYCONNECTORS.GENERICSCIM" level="[LOG_LEVEL]"
useParentHandlers="false">
     <handler name="genericscim-handler"/>
     <handler name="console-handler"/>
   </logger>
```

   b. Replace both occurrences of **[LOG_LEVEL]** with the ODL message type and level combination that you require. Table 3-9 lists the supported message type and level combinations.

      Similarly, replace **[FILE_NAME]** with the full path and name of the log file in which you want log messages specific to connector operations to be recorded.

      The following blocks show sample values for **[LOG_LEVEL]** and **[FILE_NAME]** :

```
<log_handler name='genericscim-handler' level='NOTIFICATION:1'
class='oracle.core.ojdl.logging.ODLHandlerFactory'>
<property name='logreader:' value='off'/>
     <property name='path' value='/<%OIM_DOMAIN%>/servers/oim_server1/
logs/genericScriptLogs.log>"
     <property name='format' value='ODL-Text'/>
     <property name='useThreadName' value='true'/>
     <property name='locale' value='en'/>
     <property name='maxFileSize' value='5242880'/>
     <property name='maxLogSize' value='52428800'/>
     <property name='encoding' value='UTF-8'/>
   </log_handler>
```

```
<logger name="ORG.IDENTITYCONNECTORS.GENERICSCIM" level="NOTIFICATION:1"
useParentHandlers="false">
     <handler name="genericscim-handler"/>
     <handler name="console-handler"/>
   </logger>
```

With these sample values, when you use Oracle Identity Manager, all messages generated for this connector that are of a log level equal to or higher than the `NOTIFICATION:1` level are recorded in the specified file.

2. Save and close the file.
3. Set the following environment variable to redirect the server logs to a file:
   - **For Microsoft Windows**: set WLS_REDIRECT_LOG=***FILENAME***
   - **For UNIX**: export WLS_REDIRECT_LOG=***FILENAME***

   Replace ***FILENAME*** with the location and name of the file to which you want to redirect the output.
4. Restart the application server.

3.3.5 Configuring SSL for the Generic SCIM Connector

Configure SSL to secure data communication between Oracle Identity Manager and the target system

1. Obtain the SSL public key certificate for the SCIM-based target system.
2. Copy the public key certificate of the SCIM-based target system to the computer hosting Oracle Identity Manager.
3. Run the following `keytool` command to import the target system certificate into the Oracle WebLogic Server keystore:

   `keytool -import -keystore` *`KEYSTORE_NAME`* `-storepass` *`PASSWORD`* `-file` *`CERT_FILE_NAME`* `-alias` *`ALIAS`*
   In this command:
   - *KEYSTORE_NAME* is the full path and name of the DemoTrust keystore.
   - *PASSWORD* is the password of the keystore.
   - *CERT_FILE_NAME* is the full path and name of the certificate file.
   - *ALIAS* is the target system certificate alias.

   The following is a sample value for this command:

   ```
   keytool -import -keystore WEBLOGIC_HOME/server/lib/DemoTrust.jks -
   storepass DemoTrustKeyStorePassPhrase –file /home/target.cert –alias
   serverwl
   ```

Note:

- Change the parameter values passed to the `keytool` command according to your requirements. Ensure that there is no line break in the keytool arguments
- Ensure that the system date for Oracle Identity Manager is in sync with the validity date of the SSL certificate to avoid any errors during SSL communication.

4

Using the Generic SCIM Connector

You can use the connector for performing reconciliation and provisioning operations after configuring it to meet your requirements.

This section discusses the following topics:

> **Note:**
>
> These sections provide both conceptual and procedural information about configuring the connector. It is recommended that you read the conceptual information before you perform the procedures.

- Lookup Definitions Used During Connector Operations
- Configuring Reconciliation
- Scheduled Jobs
- Performing Provisioning Operations
- Uninstalling the Connector

4.1 Lookup Definitions Used During Connector Operations

Lookup definitions used during reconciliation and provisioning are either preconfigured or can be synchronized with the target system.

The following categories of lookup definitions are discussed in this section:

- Predefined Lookup Definitions
- Lookup Definitions Synchronized with the Target System

4.1.1 Predefined Lookup Definitions

Preconfigured lookup definitions are the other lookup definitions that are created in Oracle Identity Manager when you deploy the connector. These lookup definitions are either prepopulated with values or values must be manually entered in them after the connector is deployed.

The other lookup definitions are as follows:

- Lookup.*RESOURCE*.Configuration
- Lookup.*RESOURCE*.UM.Configuration
- Lookup.*RESOURCE*.UM.ReconAttrMap
- Lookup.*RESOURCE*.UM.ProvAttrMap

- Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults

> **Note:**
>
> *RESOURCE* has been used as a place holder text for IT resource name. Therefore, replace all instances of *RESOURCE* in this guide with the value that you specified for the itResourceName entry in the GenericScimConfiguration.groovy file. See Understanding Entries in the Predefined Sections of the Groovy File for more information about entries in the GenericScimConfiguration.groovy file.

4.1.1.1 Lookup.*RESOURCE*.Configuration

The Lookup.RESOURCE.Configuration lookup definition holds connector configuration entries that are used during reconciliation (both trusted source and target resource) and provisioning operations.
Table 4-1 lists the entries in this lookup definition.

Table 4-1 Entries in the Lookup.RESOURCE.Configuration Lookup Definition

Code Key	Decode	Description
Bundle Name	org.identityconnectors.genericscim	This entry holds the name of the connector bundle class. Do *not* modify this entry.
Bundle Version	1.0.1115	This entry holds the version of the connector bundle class. Do *not* modify this entry.
Connector Name	org.identityconnectors.genericscim.GenericSCIMConnector	This entry holds the name of the connector class. Do *not* modify this entry.
User Configuration Lookup	Lookup.*RESOURCE*.UM.Configuration	This entry holds the name of the lookup definition that contains configuration information specific to the user object type. See Lookup.*RESOURCE*.UM.Configuration for more information about this lookup definition.

4.1.1.2 Lookup.*RESOURCE*.UM.Configuration

The *Lookup.RESOURCE.UM.Configuration* lookup definition contains entries specific to the user object type. This lookup definition is preconfigured.

Table 4-2 lists the default entries in this lookup definition when you have configured your target system as a target resource.

Table 4-2 Entries in the Lookup.*RESOURCE*.UM.Configuration Lookup Definition for a Target Resource Configuration

Code Key	Decode
Provisioning Attribute Map	Lookup.*RESOURCE*.UM.ProvAttrMap
Recon Attribute Map	Lookup.*RESOURCE*.UM.ReconAttrMap

Table 4-3 lists the default entries in this lookup definition when you have configured your target system as a trusted source.

Table 4-3 Entries in the Lookup.*RESOURCE*.UM.Configuration Lookup Definition for a Trusted Source Configuration

Code Key	Decode
Recon Attribute Map	Lookup.*RESOURCE*.UM.ReconAttrMap
Recon Attribute Defaults	Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults

4.1.1.3 Lookup.*RESOURCE*.UM.ReconAttrMap

The Lookup.*RESOURCE*.UM.ReconAttrMap lookup definition holds mappings between resource object fields and target system attributes

Depending on whether you have configured your connector for the target resource mode or trusted source mode, this lookup definition is used during target resource or trusted source user reconciliation runs, respectively.

If you have configured the connector for target resource mode:

The following is the format of the Code Key and Decode values in this lookup definition:

For single-valued attributes:

- **Code Key:** Reconciliation attribute of the resource object against which target resource user reconciliation runs must be performed
- **Decode:** Corresponding target system attribute name

For multivalued attributes:

- **Code Key:** *RO_ATTR_NAME~ATTR_NAME*[LOOKUP]

  In this format:

  – *RO_ATTR_NAME* specifies the reconciliation field for the child table.

  – *ATTR_NAME* is the name of the multivalued attribute.

  – [LOOKUP] is a keyword that is appended to the code key value if the child data is picked from a lookup or declared as an entitlement.

- **Decode:** Corresponding target system attribute name

  EMBED_OBJ_NAME~RELATION_TABLE_NAME~ATTR_NAME

  In this format:

  - *EMBED_OBJ_NAME* is the name of the object (for example, an account's address) on the target system that is embedded in another object.
  - *RELATION_TABLE_NAME* is the name of child table in the target system.
  - *ATTR_NAME* is the name of the column in the child table corresponding to the multivalued attribute in the Code Key column.

If you have configured your connector for trusted source mode:

The following is the format of the Code Key and Decode values in this lookup definition:

- **Code Key:** Reconciliation attribute of the resource object against which trusted source user reconciliation runs must be performed
- **Decode:** Corresponding target system attribute name

The entries in this lookup definition depend on the data available in the target system. The entries of this lookup definition are populated based on the values specified for the alias entry in the GenericScimConfiguration.groovy file. See Understanding Entries in the Predefined Sections of the Groovy File for more information about the alias entry.

4.1.1.4 Lookup.*RESOURCE*.UM.ProvAttrMap

The Lookup.*RESOURCE*.UM.ProvAttrMap lookup definition holds mappings between process form fields and target system attribute names. This lookup definition is used for performing provisioning operations.

The following is the format of the Code Key and Decode values in this lookup definition:

- **Code Key:** Name of the label on the process form
- **Decode:** Corresponding target system attribute name

For entries corresponding to child form fields, the following is the format of the Code Key and Decode values:

- **Code Key:** *CHILD_FORM_NAME~FIELD_NAME*

  In this format:

  - *CHILD_FORM_ NAME* specifies the name of the child form.
  - *FIELD_NAME* specifies the name of the label on the child form.
- **Decode:** Combination of the following elements separated by the tilde (~) character:

  EMBED_OBJ_NAME~RELATION_TABLE_NAME~COL_NAME

  In this format:

  - *EMBED_OBJ_NAME* is the name of the object (for example, an account's address) on the target system that is embedded in another object.
  - *RELATION_TABLE_NAME* is the name of child table in the target system.
  - *COL_NAME* is the name of the column in the child table corresponding to the child form specified in the Code Key column.

The entries in this lookup definition depend on the data available in the target system. The values in the lookup definition are populated based on the value specified for the

alias entry in the GenericScimConfiguration.groovy file. See Understanding Entries in the Predefined Sections of the Groovy File for more information about the alias entry.

4.1.1.5 Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults

The Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults lookup definition holds default values of the mandatory fields on the Oracle Identity Manager User form that are not mapped with the target system attributes. This lookup definition is created only if you have configured the connector for the trusted source mode. .

The Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults lookup definition is used when there is a mandatory field on the Oracle Identity Manager User form, but no corresponding attribute in the target system from which values can be fetched during trusted source reconciliation runs. In addition, this lookup definition is used if the mandatory field on the Oracle Identity Manager User form has a corresponding column that is empty or contains null values.

The following is the format of the Code Key and Decode values in this lookup definition:

- **Code Key:** Name of the user field on Oracle Identity Self Service.
- **Decode:** Corresponding default value to be displayed.

For example, the Role field is a mandatory field on the Oracle Identity Manager User form. Suppose the target system contains no attribute that stores information about the role for a user account. During reconciliation, no value for the Role field is fetched from the target system. However, as the Role field cannot be left empty, you must specify a value for this field. Therefore, the Decode value of the Role Code Key has been set to `Full-Time`. This implies that the value of the Role field on the Oracle Identity Manager User form displays Full-Time for all user accounts reconciled from the target system.

Table 4-4 lists the default entries in this lookup definition.

Table 4-4 Entries in the Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults Lookup Definition

Code Key	Decode
Role	Full-Time
Organization Name	Xellerate Users
Xellerate Type	End-User

4.1.2 Lookup Definitions Synchronized with the Target System

Lookup field synchronization involves copying additions or changes made to specific fields in the target system to lookup definitions in Oracle Identity Manager.

During a provisioning operation, you use a lookup field on the process form to specify a single value from a set of values. For example, you may want to select a role from a lookup field (displaying a set of roles) to specify the role being assigned to the user.

While configuring the GenericScimConfiguration.groovy file, if you specified a value for the lookupAttributeList entry, then the connector creates a lookup definition for every target system attribute specified in this entry and then associates it with the

corresponding lookup field on the OIM User process form. The connector creates a lookup definition named in the following format:

Lookup.${*IT_RES_NAME*}.${*FIELD_NAME*}

In this format, the connector replaces:

- *IT_RES_NAME* with the value of the itResourceDefName entry in the GenericScimConfiguration.groovy file.
- *FIELD_NAME* with the name of the field for which the lookup field is created.

Lookup field synchronization involves copying additions or changes made to the target system attributes (listed in the lookupAttributeList entry) into corresponding lookup definitions (used as an input source for lookup fields) in Oracle Identity Manager. This is achieved by running scheduled jobs for lookup field synchronization.

The following example illustrates the list of lookup definitions created for a given lookupAttributeList value:

Suppose the value of the itResourceDefName entry is GenSCIM. If the value of the lookupAttributeList entry is `['Roles', 'Groups']`, then the connector creates the following lookup definitions:

- Lookup.GenSCIM.Roles
- Lookup.GenSCIM.Groups

After you perform lookup field synchronization, data in the lookup definition is stored in the following format:

- **Code Key value:** *IT_RESOURCE_KEY~LOOKUP_FIELD_ID*

  In this format:

  – *IT_RESOURCE_KEY* is the numeric code assigned to each IT resource in Oracle Identity Manager.

  – *LOOKUP_FIELD_ID* is the target system code assigned to each lookup field entry. This value is populated based on the target system attribute name specified in the Code Key attribute of the scheduled job for lookup field synchronization.

  Sample value: `1~SA`

- **Decode value:** *IT_RESOURCE_NAME~LOOKUP_FIELD_ID*

  In this format:

  – *IT_RESOURCE_NAME* is the name of the IT resource in Oracle Identity Manager.

  – *LOOKUP_FIELD_ID* is the target system code assigned to each lookup field entry. This value is populated based on the target system attribute name specified in the Decode attribute of the scheduled job for lookup field synchronization.

  Sample value: `GenSCIM~SYS_ADMIN`

> **See Also:**
>
> Scheduled Job for Lookup Field Synchronization for information about the attributes of the scheduled job for lookup field synchronization

4.2 Configuring Reconciliation

You can configure the connector to specify the type of reconciliation and its schedule.

This section discusses the following topics related to configuring reconciliation:

- Reconciliation Rules for the Generic SCIM Connector
- Full Reconciliation and Incremental Reconciliation
- Limited Reconciliation for Generic SCIM Connector
- Lookup Field Synchronization

4.2.1 Reconciliation Rules for the Generic SCIM Connector

Reconciliation rules are automatically created when you generate the Generic SCIM connector.

The following is the format of the rule element:

```
User Login Equals NameAttribute
```

In this rule element:

- User Login is the User ID field on the Oracle Identity Manager User form.
- NameAttribute is the value of the account qualifier in the schema.properties file.

For example, if the value of the NameAttribute account qualifier is __NAME__, then the rule element is as follows:

```
User Login Equals__NAME__
```

4.2.2 Full Reconciliation and Incremental Reconciliation

Full reconciliation involves reconciling all existing user records from the target system into Oracle Identity Manager.

In **incremental reconciliation,** only records created or modified after the latest date or timestamp the last reconciliation was run are considered for reconciliation.

After you deploy the connector, you must first perform full reconciliation.

You can perform a full reconciliation by removing or deleting any value currently assigned to the Filter attribute and then running the scheduled job for user data reconciliation. See Scheduled Jobs for Reconciliation of User Records for more information about the user reconciliation scheduled job. In this scheduled job, you can include the timestamp attributes available in the Incremental Recon Attribute field.

At any given point in time, you can switch from incremental reconciliation to full reconciliation. All you need to do is perform a full reconciliation run.

To perform incremental reconciliation, you must update and run the scheduled job for user data reconciliation to include the following attributes:

- Incremental Recon Attribute — Name of the target system attribute that holds the time stamp at which the record was last modified. The value in this attribute is used to determine the newest or latest record reconciled from the target system.
- Latest Token — Holds the value of the attribute that is specified as the value of the Incremental Recon Attribute attribute. The Latest Token attribute is used for internal purposes. Do not enter a value for this attribute. The reconciliation engine automatically enters a value in this attribute. Sample value: `1354753427000`

4.2.3 Limited Reconciliation for Generic SCIM Connector

Limited or filtered reconciliation is the process of limiting the number of records being reconciled based on a set filter criteria.

By default, all target system records that are added or modified after the last reconciliation run are reconciled during the current reconciliation run. You can customize this process by specifying the subset of added or modified target system records that must be reconciled. You do this by creating filters for the reconciliation module.

You can perform limited reconciliation by creating filters that your target system supports. This connector provides the Filter attribute (scheduled task attributes) that allows you to use any of the attributes of the target system to filter target system records.

4.2.4 Lookup Field Synchronization

Lookup field synchronization involves obtaining the most current values from specific attributes in the target system to the lookup definitions (used as an input source for lookup fields) in Oracle Identity Manager. You can perform lookup field synchronization by configuring and running the scheduled jobs for lookup field synchronization.

Scheduled jobs for lookup field synchronization are created only if you have specified a value for the lookupAttributeList entry in the GenericScimConfiguration.groovy file. The names of these scheduled jobs are in the following format:

IT_RES_NAME Target *FIELD_NAME* Lookup Reconciliation

For every attribute specified in the lookupAttributeList entry, a corresponding scheduled job for reconciling lookup values from the target system is created. This is illustrated by the following example:

Suppose the value of the itResourceDefName entry is GenSCIM. If the value of the lookupAttributeList entry is `['Roles', 'Groups']`, then the connector creates the following scheduled jobs:

- GenSCIM Target Roles Lookup Reconciliation
- GenSCIM Target Groups Lookup Reconciliation

> **Note:**
>
> See Scheduled Job for Lookup Field Synchronization.

4.3 Scheduled Jobs

When you run the Connector Installer, reconciliation scheduled jobs are automatically created in Oracle Identity Manager. You must configure these scheduled jobs to suit your requirements by specifying values for its attributes.

The scheduled jobs that you can configure for reconciliation are discussed in the following sections:

- Scheduled Job for Lookup Field Synchronization
- Scheduled Jobs for Reconciliation of User Records
- Scheduled Jobs for Reconciliation of Deleted Users Records
- Configuring Scheduled Jobs

4.3.1 Scheduled Job for Lookup Field Synchronization

Scheduled jobs for lookup field synchronization fetch the most recent values from specific fields in the target system to lookup definitions in Oracle Identity Manager. These lookup definitions are used as an input source for lookup fields in Oracle Identity Manager.

After you generate the connector, scheduled jobs for lookup field synchronization are created only if you have specified a value for the lookupAttributeList entry in the GenericScimConfiguration.groovy file. For every attribute specified in the lookupAttributeList entry, a corresponding scheduled job for reconciling lookup values from the target system is created.

Table 4-5 describes the attributes of the scheduled job for lookup field synchronization. See Configuring Scheduled Jobs.

Table 4-5 Attributes of the Scheduled Job for Lookup Field Synchronization

Attribute	Description
Code Key Attribute	Enter the name of the attribute that is used to populate the Code Key column of the lookup definition (specified as the value of the Lookup Name attribute).
Decode Attribute	Enter the name of the attribute that is used to populate the Decode column of the lookup definition (specified as the value of the Lookup Name attribute).
IT Resource Name	Name of the IT resource for the target system installation from which you want to reconcile records. The default value of this attribute is the same as the value of the ITResourceDefName entry in the GenericScimConfiguration.groovy file.

Table 4-5 (Cont.) Attributes of the Scheduled Job for Lookup Field Synchronization

Attribute	Description
Lookup Name	Name of the lookup definition in Oracle Identity Manager that must be populated with values fetched from the target system. The value for this attribute is populated automatically if you have specified a value for the lookupAttributeList entry while configuring the GenericScimConfiguration.groovy file. The value of this attribute is in the following format: Lookup.${*IT_RES_NAME*}.${*FIELD_NAME*} For example, if you have specified Roles as the value of the lookupAttributeList entry, then the value of this attribute is Lookup.GenSCIMTrusted.Roles.
Object Type	Enter the type of object you want to reconcile. Default value: `OTHER` **Note:** • For lookup field synchronization, the object type must be any object other than `User`. • You must set the object type to the corresponding object value before running the scheduled job. For example, set the Object Type value to `Organization` if you want to run the Organization lookup schedule job.

4.3.2 Scheduled Jobs for Reconciliation of User Records

After you generate the connector, the scheduled task for user data reconciliation is automatically created in Oracle Identity Manager. A scheduled job, which is an instance of this scheduled task is used to reconcile user data from the target system.

The following scheduled jobs are used for user data reconciliation:

- *RESOURCE* Target Resource User Reconciliation

  This scheduled job is used to reconcile user data in the target resource (account management) mode of the connector.

- *RESOURCE* Trusted Resource User Reconciliation

  This scheduled job is used to reconcile user data in the trusted source (identity management) mode of the connector.

Table 4-6 describes the attributes of both scheduled jobs.

Table 4-6 Attributes of the User Reconciliation Scheduled Jobs

Attribute	Description
Filter	Enter the search filter for fetching records from the target system during a reconciliation run. See Limited Reconciliation for Generic SCIM Connector.
Incremental Recon Attribute	Enter the name of the target system attribute that holds the time stamp at which the record was last modified. The value in this attribute is used during incremental reconciliation to determine the newest or latest record reconciled from the target system.
Latest Token	This attribute holds the value of the attribute that is specified as the value of the Incremental Recon Attribute attribute. The Latest Token attribute is used for internal purposes. By default, this value is empty. **Note**: Do not enter a value for this attribute. The reconciliation engine automatically enters a value in this attribute.
IT Resource Name	Name of the IT resource for the target system installation from which you want to reconcile user records. Sample value: `GenSCIM`
Object Type	Type of object you want to reconcile. Default value: `User` **Note:** User is the only object that is supported. Therefore, do not change the value of this attribute.
Resource Object Name	Name of the resource object that is used for reconciliation. Sample value: `GenSCIM User`
Scheduled Task Name	Name of the scheduled task that is used for reconciliation. The default value of this attribute in the `RESOURCE Target Resource User Reconciliation` scheduled job is `RESOURCE Target Resource User Reconciliation`. The default value of this attribute in the `RESOURCE Trusted Resource User Reconciliation` scheduled job is `RESOURCE Trusted Resource User Reconciliation`. Sample value: `User Target Reconciliation`

4.3.3 Scheduled Jobs for Reconciliation of Deleted Users Records

After you generate the connector, the scheduled task for reconciling data about deleted users records is automatically created in Oracle Identity Manager. A scheduled

job, which is an instance of this scheduled task is used to reconcile data about deleted users in the target system.

The following scheduled jobs are used for reconciliation of deleted user records data:

- *RESOURCE* Target Resource User Delete Reconciliation

  This scheduled job is used to reconcile data about deleted user records in the target resource (account management) mode of the connector. During a reconciliation run, for each deleted user record on the target system, the target system resource is revoked for the corresponding Oracle Identity Manager User.

- *RESOURCE* Trusted User Delete Reconciliation

  This scheduled job is used to reconcile data about deleted user records in the trusted source (identity management) mode of the connector. During a reconciliation run, for each deleted target system user record, the corresponding Oracle Identity Manager User is deleted.

Table 4-7 describes the attributes of both scheduled jobs.

Table 4-7 Attributes of the Delete User Reconciliation Scheduled Jobs

Attribute	Description
IT Resource Name	Name of the IT resource for the target system installation from which you want to reconcile user records. Sample value: `GenSCIMTrusted`
Object Type	Type of object you want to reconcile. Default value: `User` **Note:** User is the only object that is supported. Therefore, do not change the value of this attribute.
Resource Object Name	Name of the resource object that is used for delete reconciliation.

4.3.4 Configuring Scheduled Jobs

Configure scheduled jobs to perform reconciliation runs that check for new information on your target system periodically and replicates the data in Oracle Identity Manager.

To configure a scheduled job:

1. Log in to Oracle Identity System Administration.
2. In the left pane, under System Management, click **Scheduler.**
3. Search for and open the scheduled task as follows:
   a. On the left pane, in the Search field, enter the name of the scheduled job as the search criterion. Alternatively, you can click **Advanced Search** and specify the search criterion.
   b. In the search results table on the left pane, click the scheduled job in the Job Name column.
4. On the Job Details tab, you can modify the following parameters:

- **Retries:** Enter an integer value in this field. This number represents the number of times the scheduler tries to start the job before assigning the Stopped status to the job.
- **Schedule Type:** Depending on the frequency at which you want the job to run, select the appropriate schedule type.

In addition to modifying the job details, you can enable or disable a job.

5. On the Job Details tab, in the Parameters region, specify values for the attributes of the scheduled task.

> **Note:**
>
> - Attribute values are predefined in the connector XML file that you import. Specify values only for those attributes that you want to change.
> - Values (either default or user-defined) must be assigned to all the attributes. If even a single attribute value is left empty, then reconciliation is not performed.
> - Attributes of the scheduled job are discussed in Scheduled Jobs.

6. Click **Apply** to save the changes.

> **Note:**
>
> You can use the Scheduler Status page in Identity System Administration to either start, stop, or reinitialize the scheduler.

4.4 Performing Provisioning Operations

You create a new user in Oracle Identity Self Service by using the Create User page. You provision or request for accounts on the Accounts tab of the User Details page.

To perform provisioning operations in Oracle Identity Manager:

1. Log in to Oracle Identity System Administration.
2. Create a user as follows:
   a. In Identity Self Service, click **Manage**. The Home tab displays the different Manage option. Click **Users**. The Manage Users page is displayed.
   b. From the Actions menu, select **Create**. Alternatively, you can click **Create** on the toolbar. The Create User page is displayed with input fields for user profile attributes.
   c. Enter details of the user in the Create User page.
3. On the Account tab, click **Request Accounts.**
4. In the Catalog page, search for and add to cart the application instance created for the IT resource (in Associating the Form with the Application Instance), and then click **Checkout**.

> **Note:**
>
> Ensure to select proper values for lookup type fields as there are a few dependent fields. Selecting a wrong value for such fields may result in provisioning failure.

5. Click Ready to **Submit.**
6. Click **Submit.**
7. If you want to provision entitlements, then:
   a. On the Entitlements tab, click **Request Entitlements.**
   b. In the Catalog page, search for and add to cart the entitlement, and then click **Checkout.**
   c. Click **Submit.**

4.5 Uninstalling the Connector

Uninstalling the connector deletes all the account related data associated with resource objects of the connector. You use the Uninstall Connectors utility to uninstall a connector.

If you want to uninstall the connector for any reason, see Uninstalling Connectors in *Oracle Fusion Middleware Administering Oracle Identity Manager*.

5

Extending the Functionality of the Generic SCIM Connector

After you generate and install the connector, you can configure it to address your specific business requirements.

This chapter provides information about the following optional configuration procedures:

- Adding Custom OIM User Fields for Trusted Source Reconciliation
- Adding Custom Fields for Target Resource Reconciliation
- Adding Custom Fields for Provisioning
- Configuring Transformation of Data During User Reconciliation
- Configuring Validation of Data During Reconciliation and Provisioning

5.1 Adding Custom OIM User Fields for Trusted Source Reconciliation

While generating the connector, you create mappings between Oracle Identity Manager User fields and the corresponding target system fields by specifying a value for the alias entry. After generating the connector, if there are additional target system fields that you want to use during trusted source reconciliation, then you can extend the set of fields by creating custom or user-defined fields (UDFs).

To add new fields for trusted source reconciliation:

1. Add the new field on the Oracle Identity Manager User process form. See Configuring Custom Attributes in *Oracle Fusion Middleware Administering Oracle Identity Manager* for information on creating UDFs.

   > **Note:**
   >
   > If the new field that you want to add is already present on the Oracle Identity Manager User field, then skip this step and proceed to the next step.

2. Log in to the Design Console.
3. In the resource object definition, add the reconciliation field corresponding to the attribute as follows:
   a. Expand the **Resource Management** folder, and then double-click **Resource Objects.**
   b. Search for and open the resource object corresponding to your target system.

c. On the Object Reconciliation tab, click **Add Field** to open the Add Reconciliation Field dialog box.

d. Specify a value for the field name. For example, `Building.`

e. From the Field Type list, select a data type for the field. In addition, if you want to designate the attribute as a mandatory attribute, then select the check box.

f. Click the Save icon, and then close the dialog box.

g. Click the Save icon.

4. Create a reconciliation field mapping in the process definition as follows:

a. Expand the **Process Management** folder, and then double-click **Process Definition.**

b. Search for and open the process definition for your target system.

c. On the Reconciliation Field Mapping tab, click **Add Field Map.**

d. From the Field Name list in the Add Reconciliation Field Mapping dialog box, select the name that you have assigned to the attribute created in the resource object.

e. Select a value from the **User Attribute** menu and click **OK.**

f. If the field mapping is a key field for matching the process data, check the key Field for Reconciliation matching check box.

g. Click the Save icon.

5. Create a reconciliation profile as follows:

a. Expand the **Resource Management** folder, and then double-click **Resource Objects.**

b. Search for and open the resource object corresponding to your target system.

c. On the Object Reconciliation tab, click **Create Reconciliation Profile.** This copies changes made to the resource object into the MDS.

d. Click the Save icon.

6. Add an entry for the attribute in the lookup definition for reconciliation attribute mapping as follows:

a. Expand the **Administration** folder, and then double-click **Lookup Definition.**

b. Search for and open the Lookup.*RESOURCE*.UM.ReconAttrMap lookup definition.

c. To add a roe, click **Add.**

d. In the **Code Key** column, enter the name that you have set for the attribute in the resource object. For example, `Building.`

e. In the **Decode** column, enter the corresponding name of the target system column. For example, `BUILDING.`

f. Click the Save icon.

5.2 Adding Custom Fields for Target Resource Reconciliation

While generating the connector, you create mappings between Oracle Identity Manager User fields and the corresponding target system fields by specifying a value for the alias entry. After generating the connector, if there are additional target system fields that you want to use during target resource reconciliation, then you can extend the set of fields by creating custom or user-defined fields (UDFs). See Creating a Custom Attribute in *Oracle Fusion Middleware Administering Oracle Identity Manager*.

To add a custom field for reconciliation:

1. Log in to the Design Console.
2. In the resource object definition, add the reconciliation field corresponding to the attribute as follows:
   a. Expand the **Resource Management** folder, and then double-click **Resource Objects.**
   b. Search for and open the resource object corresponding to your target system.
   c. On the Object Reconciliation tab, click **Add Field** to open the Add Reconciliation Field dialog box.
   d. Specify a value for the field name. For example, `Building`.
   e. From the Field Type list, select a data type for the field. In addition, if you want to designate the attribute as a mandatory attribute, then select the check box.
   f. Click the Save icon, and then close the dialog box.
   g. Click the Save icon.
3. Add an entry for the attribute in the lookup definition for reconciliation attribute mapping as follows:
   a. Expand the **Administration** folder, and then double-click **Lookup Definition**.
   b. Search for and open the *Lookup.RESOURCE.UM.ReconAttrMap* lookup definition.
   c. To add a row, click **Add.**
   d. In the **Code Key** column, enter the name that you have set for the attribute in the resource object. For example, `Building`.
   e. In the **Decode** column, enter the corresponding name of the target system column. For example, `BUILDING`.
   f. Click the Save icon.
4. Add the attribute as a field on the process form as follows:
   a. Expand the **Development Tools** folder, and then double-click **Form Designer**.
   b. Search for and open the process form for your target system.
   c. Click **Create New Version** to create a version of the process form. Then, enter a version name and click the Save icon.

d. Click **Add**.

e. In the newly added row, enter values for the Name, Variant Type, Field Label, and Field Type columns. If required, enter values for the rest of the columns.

> **Note:**
>
> - If the attribute on the target system is of the Time, or Timestamp format, then set the value of the Variant Type column to **String**.
> - If you want to handle date attributes of the target system as a date editor, then set the value of the Variant Type column to **Date**. Otherwise, set it to **String**.

f. Click the Save icon.

g. Click **Make Version Active** to activate the new version of the process form.

5. Create a reconciliation field mapping in the process definition as follows:

   a. Expand the **Process Management** folder, and then double-click **Process Definition**.

   b. Search for and open the process definition for your target system.

   c. On the Reconciliation Field Mapping tab, click **Add Field Map**.

   d. From the Field Name list in the Add Reconciliation Field Mapping dialog box, select the name that you have assigned to the attribute created in the resource object.

   e. Double-click the Process Data Field, a new pop-up will appear. The entries in the pop-up correspond to the process form fields.

   f. Select the corresponding newly added field from the pop-up.

   g. If the field mapping is a key field for matching the process data, check the key Field for Reconciliation matching check box.

   h. Click the Save icon.

6. Create a reconciliation profile as follows:

   a. Expand the **Resource Management** folder, and then double-click **Resource Objects**.

   b. Search for and open the resource object corresponding to your target system.

   c. On the Object Reconciliation tab, click **Create Reconciliation Profile**. This copies changes made to the resource object into the MDS.

   d. Click the Save icon.

7. Perform all changes made to the Form Designer of the Design Console in a new UI form as follows:

   a. Log in to Oracle Identity System Administration.

   b. Create and active a sandbox. See Creating and Activating a Sandbox.

   c. Create a new UI form to view the newly added field along with the rest of the fields. See Creating a New UI Form.

d. Associate the newly created UI form with the application instance of your target system. To do so, open the existing application instance for your resource, from the Form field, select the form (created in Step 7.c), and then save the application instance.

e. Publish the sandbox. See Publishing a Sandbox.

8. Add the attribute for provisioning. See Adding Custom Fields for Provisioning.

5.3 Adding Custom Fields for Provisioning

While generating the connector, by performing the procedure described in Generating the Generic SCIM Connector, you create mappings between the Oracle Identity Manager User fields and the corresponding target system fields (columns) by specifying a value for the alias entry. If there are additional target system fields that you want to use during provisioning, then you can extend the existing set of fields by creating custom or user-defined fields (UDFs).

To add a new user-defined field for provisioning:

1. Add the attribute as a field on the process form as follows:

> **Note:**
>
> Directly proceed to the next step if you have already added the field to the process form while performing the procedure described in Adding Custom Fields for Target Resource Reconciliation.

a. Expand **Development Tools**, and then double-click **Form Designer**.

b. Search for and open the process form for your target system.

c. Click **Create New Version** to create a version of the form. Then, enter a version name and click the Save icon.

d. Click **Add**.

e. In the newly added row, enter values for the Name, Variant Type, Field Label, and Field Type columns. If required, enter values for the rest of the columns.

> **Note:**
>
> - If the attribute on the target system is of the Time, or Timestamp format, then set the value of the Variant Type column to **String**.
> - If you want to handle date attributes of the target system as a date editor, then set the value of the Variant Type column to **Date**. Otherwise, set it to **String**.

f. Click the Save icon.

g. Click **Make Version Active** to activate the new version of the process form.

2. Perform all changes made to the Form Designer of the Design Console (in Step 1) in a new UI form as follows:

a. Log in to Oracle Identity System Administration.

b. Create and active a sandbox. See Creating and Activating a Sandbox .

c. Create a new UI form to view the newly added field along with the rest of the fields. See Creating a New UI Form.

d. Associate the newly created UI form with the application instance of your target system. To do so, open the existing application instance for your resource, from the Form field, select the form (created in Step 2.c), and then save the application instance.

e. Publish the sandbox. See Publishing a Sandbox.

3. Add an entry in the lookup definition for provisioning attribute mappings as follows:

a. Expand **Administration**, and then double-click **Lookup Definition**.

b. Search for and open the *Lookup.RESOURCE.UM.ProvAttrMap* lookup definition.

c. To add a row, click **Add**.

d. In the **Code Key** column, enter the field label for the attribute on the process form. See Step 1 for information about this field name.

e. In the **Decode** column, enter the corresponding name of the target system column. For example, `BUILDING`.

f. Click the Save icon.

4. To enable updates of the attribute, add an update process task in the process definition as follows:

a. Expand **Process Management**, and then double-click **Process Definition**.

b. Search for and open the process definition for your target system.

c. On the Tasks tab, click **Add**.

d. On the General tab of the dialog box that is displayed, enter a name and description for the task, and then select the following fields in the Task Properties section:

- Conditional
- Required for Completion
- Allow Cancellation while Pending
- Allow Multiple Instances

> **Note:**
>
> The name must be in the *PROCESS_FORM_FIELD_NAME* Updated format.

e. Click the Save icon.

f. On the Integration tab, attach the adapter responsible for performing the update account provisioning operations and map the adapter variables as listed in the following table:

Variable Name	Data Type	Map To	Qualifier	Literal Value
processKeyInstance	Long	Process Data	Process Instance	NA
Adapter return value	Object	Response Code	NA	NA
objectType	String	Literal	String	User
attrFieldName	String	Literal	String	Building
itResourceFieldName	String	Literal	String	IT Resource Form Field Name

g. Click the Save icon.

h. On the Response tab, add appropriate responses.

i. Click the Save icon.

j. Click the Save icon and then close the dialog box.

5. Adding the attribute for reconciliation.

   When you add an attribute on the process form, you must also enable reconciliation of values for that attribute from the target system. See Adding Custom Fields for Target Resource Reconciliation.

5.4 Configuring Transformation of Data During User Reconciliation

You can configure transformation of reconciled single-valued data to suit your requirements. The below section describes an optional procedure. Perform this procedure only if you want to configure transformation of data during reconciliation.

As an example, you can use First Name and Last Name values to create a value for the Full Name field in Oracle Identity Manager.

To configure transformation of data:

1. Write code that implements the required transformation logic in a Java class.

   The following sample transformation class creates a value for the Full Name attribute by using values fetched from the FIRST_NAME and LAST_NAME columns of the target system:

```
package oracle.iam.connectors.common.transform;

import java.util.HashMap;

public class TransformAttribute {

        /*
        Description:Abstract method for transforming the
attributes

        param hmUserDetails<String,Object>

        HashMap containing parent data details
```

```
          param hmEntitlementDetails <String,Object>

          HashMap containing child data details

          */
          public Object transform(HashMap hmUserDetails, HashMap
hmEntitlementDetails,String sField) {
          /*
           * You must write code to transform the
attributes.
           Parent data attribute values can be fetched by
           using hmUserDetails.get("Field Name").
           *To fetch child data values, loop through the
           * ArrayList/Vector fetched by
hmEntitlementDetails.get("Child          Table")
           * Return the transformed attribute.
           */
          String sFirstName= (String)hmUserDetails.get("First
Name");
          String sLastName= (String)hmUserDetails.get("Last
Name");
          String sFullName=sFirstName+"."+sLastName;
          return sFullName;
          }
   }
```

2. Create a JAR file to hold the Java class.
3. Run the Oracle Identity Manager Upload JARs utility to post the JAR file to the Oracle Identity Manager database. This utility is copied into the following location when you install Oracle Identity Manager:

> **Note:**
>
> Before you use this utility, verify that the WL_HOME environment variable is set to the directory in which Oracle WebLogic Server is installed.

- **For Microsoft Windows:** *OIM_HOME*/server/bin/UploadJars.bat
- **For UNIX**:*OIM_HOME*/server/bin/UploadJars.sh

When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being uploaded, and the location from which the JAR file is to be uploaded. Specify 1 as the value of the JAR type.

> **See Also:**
>
> Upload JAR Utility in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager*.

4. Create a lookup definition for transformation and add an entry to it as follows:
   a. Log in to the Design Console.
   b. Expand **Administration**, and then double-click **Lookup Definition**.
   c. In the Code field, enter `Lookup.RESOURCE.UM.ReconTransformation` as the name of the lookup definition.
   d. Select the **Lookup Type** option.
   e. On the Lookup Code Information tab, click **Add**.

      A new row is added.
   f. In the **Code Key** column, enter the name of the resource object field into which you want to store the transformed value. For example: `FirstName`.
   g. In the **Decode** column, enter the name of the class that implements the transformation logic. For example, `oracle.iam.connectors.common.transform.TransformAttribute`.
   h. Save the changes to the lookup definition.
5. Add an entry in the Lookup.*RESOURCE*.UM.Configuration lookup definition to enable transformation as follows:
   a. Expand **Administration**, and then double-click **Lookup Definition**.
   b. Search for and open the **Lookup.*RESOURCE*.UM.Configuration** lookup definition.
   c. Create an entry that holds the name of the lookup definition used for transformation as follows:

      Code Key: `Recon Transformation Lookup`

      Decode: `Lookup.RESOURCE.UM.ReconTransformation`
   d. Save the changes to the lookup definition.

5.5 Configuring Validation of Data During Reconciliation and Provisioning

You can configure validation of reconciled and provisioned single-valued data according to your requirements. For example, you can validate data fetched from the FIRST_NAME column to ensure that it does not contain the number sign (#). In addition, you can validate data entered in the First Name field on the process form so that the number sign (#) is not sent to the target system during provisioning operations.

For data that fails the validation check, the following message is displayed or recorded in the log file:

```
oracle.iam.connectors.icfcommon.recon.SearchReconTask : handle : Recon
event skipped, validation failed [Validation failed for attribute:
[FIELD_NAME]]
```

> **Note:**
>
> This feature cannot be applied to the Locked/Unlocked status attribute of the target system.

To configure validation of data:

1. Write code that implements the required validation logic in a Java class.

   The following sample validation class checks if the value in the First Name attribute contains the number sign (#):

```
package com.validate;
import java.util.*;
public class MyValidation {
public boolean validate(HashMap hmUserDetails,
              HashMap hmEntitlementDetails, String field)
{
            /*
         * You must write code to validate attributes.
Parent
         * data values can be fetched by using
hmUserDetails.get(field)
         * For child data values, loop through the
         * ArrayList/Vector fetched by
hmEntitlementDetails.get("Child Table")
         * Depending on the outcome of the validation
operation,
         * the code must return true or false.
         */
         /*
         * In this sample code, the value "false" is returned if
the field
         * contains the number sign (#). Otherwise, the value
"true" is
         * returned.
         */
            boolean valid=true;
            String sFirstName=(String)
hmUserDetails.get(field);
            for(int i=0;i<sFirstName.length();i++){
              if (sFirstName.charAt(i) == '#'){
                    valid=false;
                    break;
              }
            }
            return valid;
       }
}
```

2. Create a JAR file to hold the Java class.
3. Run the Oracle Identity Manager Upload JARs utility to post the JAR file to the Oracle Identity Manager database. This utility is copied into the following location when you install Oracle Identity Manager:

> **Note:**
>
> Before you use this utility, verify that the WL_HOME environment variable is set to the directory in which Oracle WebLogic Server is installed.

- **For Microsoft Windows**:*OIM_HOME*/server/bin/UploadJars.bat
- **For UNIX**: *OIM_HOME*/server/bin/UploadJars.sh

When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being uploaded, and the location from which the JAR file is to be uploaded. Specify 1 as the value of the JAR type.

> **See Also:**
>
> Upload JAR Utility in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager*.

4. If you created the Java class for validating a process form field for reconciliation, then:
   a. Log in to the Design Console.
   b. Expand **Administration**, and then double-click **Lookup Definition**.
   c. In the Code field, enter `Lookup.RESOURCE.UM.ReconValidation` as the name of the lookup definition.
   d. Select the **Lookup Type** option.
   e. On the Lookup Code Information tab, click **Add**.

      A new row is added.
   f. In the Code Key column, enter the resource object field name. For example, `First Name`.
   g. In the Decode column, enter the class name. For example, `com.validate.MyValidation`.
   h. Save the changes to the lookup definition.
   i. Search for and open the **Lookup.*RESOURCE*.UM.Configuration** lookup definition.
   j. Create an entry with the following values:

      Code Key: `Recon Validation Lookup`

      Decode: `Lookup.RESOURCE.UM.ReconValidation`
   k. Save the changes to the lookup definition.
5. If you created the Java class for validating a process form field for provisioning, then:
   a. Log in to the Design Console.
   b. Expand **Administration**, and then double-click **Lookup Definition**.

c. In the Code field, enter `Lookup.RESOURCE.UM.ProvValidation` as the name of the lookup definition.

d. Select the **Lookup Type** option.

e. On the Lookup Code Information tab, click **Add**.

   A new row is added.

f. In the **Code Key** column, enter the process form field name. In the **Decode** column, enter the class name.

g. Save the changes to the lookup definition.

h. Search for and open the **Lookup.*RESOURCE*.UM.Configuration** lookup definition.

i. Create an entry with the following values:

   Code Key: `Provisioning Validation Lookup`

   Decode: `Lookup.RESOURCE.UM.ProvValidation`

j. Save the changes to the lookup definition.

A

Files and Directories of the Generic SCIM Connector

This appendix lists the tables that describe the files and directories corresponding to the Generic SCIM connector.

Table A-1 describes the files and directories on the installation media.

Table A-1 Files and Directories on the Connector Installation Media

File in the Installation Media Directory	Description
/bundle/ org.identityconnectors.genericscim-1.0.1115.jar	This JAR file is the ICF connector bundle.
configuration/GenericSCIM-CI.xml	This XML file contains configuration information. The Connector Installer uses this XML file to create connector components.
javadoc	This directory contains information about the Java APIs used by the connector.
metadata-generator/bin/ GenericSCIMGenerator.cmd	This file contains commands to run the metadata generator.
metadata-generator/bin/ GenericSCIMGenerator.sh	Note that the .cmd file is the Microsoft Windows version of the metadata generator. Similarly, the .sh file is the UNIX version of the metadata generator.
metadata-generator/bin/classpath.cmd metadata-generator/bin/classpath-append.cmd	These files contain the commands that add the JAR files (located in the lib directory) to the classpath on Microsoft Windows.
metadata-generator/bin/logging.properties	This file contains the default logging configurations of the metadata generation utility.
metadata-generator/lib/connector-framework-internal.jar	This JAR files contains class files that implement ICF.
metadata-generator/lib/connector-framework.jar	This JAR file contains class files that define the ICF Application Programming Interface (API). This API is used communicate between Oracle Identity Manager and this connector.
metadata-generator/lib/genericSCIM-oim-integration.jar	This JAR file contains the class files of the metadata generation utility.
metadata-generator/lib/groovy-all.jar	This JAR file contains the groovy libraries required for running the metadata generator.
metadata-generator/lib/ org.identityconnectors.genericscim-1.0.11150.jar	This JAR file is the ICF connector bundle. This file is used during metadata generation.

Table A-1 (Cont.) Files and Directories on the Connector Installation Media

File in the Installation Media Directory	Description
metadata-generator/resources/ GenericSCIMConfiguration.groovy	This file contains properties that store basic information about the target system schema, which is used to configure the mode (trusted source or target resource) in which you want to run the connector. In addition, it stores information about the manner in which the connector must connect to the target system.

Table A-2 describes the files and directories in the generated connector package.

Table A-2 Files and Directories in the Generated Connector Package

File in the Connector Package	Description
configuration/*IT_RES_DEF*-CI.xml	This XML file contains configuration information that is used by the Connector Installer during the connector installation process.
resources/genericscim-generator.properties	This property file contains locale-specific properties. You can use this file as a template to add or update locale-related properties.
xml/*IT_RES_DEF*-ConnectorConfig.xml file	This XML file contains definitions for connector components such as IT resource, lookup definitions, scheduled tasks, process forms, and resource objects. This file is also referred to as the connector configuration file.

Index