Oracle® Cloud

Using the Oracle Utilities Adapter with Oracle Integration 3

F45596-12
February 2025

Oracle Cloud Using the Oracle Utilities Adapter with Oracle Integration 3,

F45596-12

Primary Author: Oracle Corporation

Contents

Preface

The Oracle Utilities Adapter helps you build integrations with Oracle Utilities applications. These integrations can be between Oracle Utilities applications or between Oracle Utilities applications and any other applications such as ERP or Oracle Field Service (OFS).

This guide includes information and procedures to help you configure the Oracle Utilities Adapter as a connection in an Oracle Integration integration.

> **Note:**
>
> The use of this adapter may differ depending on the features you have, or whether your instance was provisioned using Standard or Enterprise edition. These differences are noted throughout this guide.

Topics:

- Audience
- Documentation Accessibility
- Diversity and Inclusion
- Related Resources
- Conventions

Audience

This guide is intended for developers who want to use this adapter in integrations in Oracle Integration.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `https://www.oracle.com/corporate/accessibility/`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `https://support.oracle.com/portal/` or visit `Oracle Accessibility Learning and Support` if you are hearing impaired.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Resources

See these Oracle resources:

- Oracle Cloud at `http://cloud.oracle.com`
- *Using Integrations in Oracle Integration 3*
- *Using the Oracle Mapper with Oracle Integration 3*
- Oracle Integration documentation on the Oracle Help Center.

Conventions

The following text conventions are used in this document:

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

1

Understand the Oracle Utilities Adapter

You can use the Oracle Utilities Adapter to integrate Oracle Utilities applications with your sales, human resources, customer service, contact center, marketing, and reporting applications. The Oracle Utilities Adapter supports web service standards for the creation of open and reusable service-oriented applications (SOA).

Topics:

The following terms are used in this guide:

Term	Description
OUAF	Oracle Utilities Application Framework
NMS or non-OUAF	Network Management System
IDCS	Oracle Identity Cloud Service

Oracle Utilities Adapter Capabilities

The Oracle Utilities Adapter lets you integrate the Oracle Utilities application suite with other Oracle applications such as Oracle Enterprise Resource Planning (ERP) and Oracle Field Service (OFS).

The Oracle Utilities Adapter provides the following capabilities:

- Provides trigger (inbound) and invoke (outbound) support. This functionality enables other applications to trigger an integration in Oracle Integration or invoke an Oracle Utilities application using web services from Oracle Integration.
- Exposes both inbound and outbound services using the Oracle Utilities service catalog. This catalog provides a simplified user experience to create data mappings at design time while exposing inbound and outbound web services of the utilities applications.
- Exposes every inbound and outbound service structure using a SOAP-based WSDL or REST OpenAPI URL-based connection.
- Integrates with on-premises Oracle Utilities applications using the on-premises connectivity agent.
- Supports connecting to private resources that are in your virtual cloud network (VCN) with a private endpoint. See Connect to Private Resources in *Provisioning and Administering Oracle Integration 3* and Configure the Endpoint Access Type.
- Supports private endpoints for both SOAP and REST endpoints for invoke and trigger connections.

- Enables you to implement the following message exchange patterns on inbound SOAP and inbound REST endpoints:
  - Synchronous request/response
  - One-way request

  See Asynchronous Trigger Support in Orchestrated Integrations.

SOAP and Secure WSDL Support

The Oracle Utilities Adapter provides secure WSDL support. You can create SOAP-based integrations using the Oracle Utilities Adapter with a SOAP catalog of inbound/outbound services exposed by an OUAF application.

- The Oracle Utilities Adapter works with a SOAP catalog exposed in the Oracle Utilities Application Framework (OUAF) in the cloud using a SOAP proxy.
- SOAP proxy and OUAF changes for the cloud are made so that the behavior of the SOAP catalog is similar as in on-premises environments except for the following important changes:
  - The WSDL to retrieve the catalog is secured by default. Therefore, the credentials must be passed to retrieve the WSDL.
  - Individual WSDLs of all services exposed by the SOAP catalog are secured by default. Therefore, credentials must be passed to retrieve the WSDL.
  - The WSDL link used to retrieve the catalog and individual WSDLs is different. It points to the SOAP proxy server. For example:

    ```
    https://host:port/soap/api/iws/ServiceCatalog?WSDL
    ```

  - The endpoint within the WSDLs also points to the SOAP proxy. For example:

    ```
    https://host:port/soap/api/iws/ServiceCatalog
    ```

- The Oracle Utilities Adapter supports use of the following security policies for SOAP:
  - Username Password Token
  - Basic Authentication
  - OAuth Client Credentials (cloud and on-premises)
  - OAuth using JWT User Assertion (only for nonagent connections)
  - OAuth Client Credentials using JWT Client Assertion (only for nonagent connections

Whenever you use the secured/protected WSDL from a cloud environment, ensure that the security policy for SOAP-based integrations is either Basic Authentication or OAuth Client Credentials.

See Configure Connection Security.

REST Support

The Oracle Utilities Adapter provides REST support. You can create REST-based integrations using the Oracle Utilities Adapter with an OpenAPI URL-based catalog of inbound/outbound services exposed by an OUAF and non-OUAF (NMS) application.

- Using the Oracle Utilities Adapter as an invoke connection in an integration invokes the inbound OUAF or non-OUAF (NMS) REST web services.

- Using the Oracle Utilities Adapter as a trigger connection in an integration consumes an outbound message from an OUAF or non-OUAF (NMS) application.
- The Oracle Utilities Adapter consumes inbound and outbound REST-based services that are available as part of an OpenAPI URL provided by OUAF and non-OUAF (NMS) applications.
- The Oracle Utilities Adapter supports REST service versions 1.0 and 2.0 of inbound web services.
- The Oracle Utilities Adapter supports use of the following security policies for REST:
  - Basic Authentication
  - OAuth Resource Owner (only for invoke and nonagent based connections)
  - OAuth Client Credentials
- Only a JSON payload is supported. An XML payload is not supported.
- Support is provided for HTTP methods GET, PUT, POST and PATCH.

Note:

The Oracle Utilities Adapter supports REST services on Oracle Utilities Application Framework (OUAF) version 4.5 or greater.
Support for Swagger 2.0 is removed in Oracle Utilities Adapter version 24.04.0 or higher. Use OpenAPI 3.*x* catalogs for connections. See Using the Swagger 2.0 REST Catalog with Oracle Utilities Adapter Version 24.04.0 or Higher.

- Catalog format for an OUAF application.

  ```
  https://host:port/ouaf/rest/ouaf/openapi/iws/catalog
  ```

  or

  ```
  https://host:port/../../../rest/openapi/iws/catalog
  ```

- Catalog format for a non-OUAF (NMS) application.

  ```
  https://host:port/nms-application_code/rest/v1/catalog
  ```

  Where:

  ```
  ###application_code is code for application. Please refer to non-OUAF
  application documentation for the application code.
  ```

OAuth 2.0 Support

The Oracle Utilities Adapter supports the Open Authorization (OAuth 2.0) security policy for REST- and SOAP-based connections.

This support enables you to configure the Oracle Utilities Adapter to consume an OpenAPI 3.*x* API for REST and secured and nonsecured WSDLs for SOAP protected with OAuth 2.0. This policy is useful when the Basic Authentication security policy does not meet your security needs.

> **Note:**
>
> The OAuth Resource Owner Password Credentials policy is supported only for REST nonagent-based connections.

See Configure Connection Security.

JWT Client and User Assertions Support

JWT client and user assertions with OAuth Client Credentials are supported with the following security policies:

- OAuth Client Credentials using JWT Client Assertion security policy
- OAuth using JWT User Assertion security policy

JWT assertions enable you to invoke a service provider that does not regard an OAuth client secret as secure. Trust is established with a key-pair exchange instead of a client secret.

Oracle Utilities Adapter Restrictions

Note the following Oracle Utilities Adapter restrictions.

- The Oracle Utilities Adapter can only be used with Oracle Utilities applications that support web services. If you are using DB file or Java Message Service (JMS) integration services, generic Oracle Integration adapters must be used and not the Oracle Utilities Adapter.
- The Resource Owner Password Credentials OAuth grant type is currently not supported for REST in an agent-based connection and SOAP-based connection.
- REST connection support for Swagger 2.0-based catalogs is removed from Oracle Utilities Adapter version 24.04.0 or higher. Only OpenAPI 3.*x* catalogs can be used.
- REST Version 2.0 Services are only supported on Oracle Utilities Application Framework (OUAF) version 4.5 or higher.
- For cloud-based connections, the Oracle Utilities Adapter can only be used with the Oracle Utilities application's cloud catalog when using a secured certificate with Oracle Integration.
- The OAuth using JWT User Assertion security policy and OAuth Client Credentials using JWT Client Assertion security policy are only supported for REST cloud-based connections.

> **Note:**
>
> There are overall service limits for Oracle Integration. A service limit is the quota or allowance set on a resource. See Service Limits.

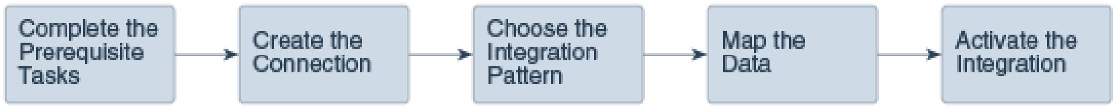
What Application Version Is Supported?

For information about which application version is supported by this adapter, see the Connectivity Certification Matrix.

Workflow to Create and Add an Oracle Utilities Adapter to an Integration

You can set up the Oracle Utilities Adapter by completing the tasks listed in the table.

This image represents the workflow for setting up an adapter:


Complete the Prerequisite Tasks
Create the Connection
Choose the Integration Pattern
Map the Data
Activate the Integration

Step	Description	More Information
1	Create the adapter connections for the applications you want to integrate. The connections can be reused in multiple integrations and are typically created by the administrator.	Create a Connection
2	Create the integration. When you do this, you add trigger and invoke connections to the integration.	Create Integrations and Add the Oracle Utilities Adapter Connection to an Integration
3	Map data between the trigger connection data structure and the invoke connection data structure.	Map Data in *Using Integrations in Oracle Integration 3*
4	(Optional) Create lookups that map the different values used by those applications to identify the same type of object (such as gender codes or country codes).	Manage Lookups in *Using Integrations in Oracle Integration 3*
5	Activate the integration.	Manage Integrations in *Using Integrations in Oracle Integration 3*
6	Monitor the integration on the dashboard.	Monitor Integrations During Runtime in *Using Integrations in Oracle Integration 3*
7	Track payload fields in messages during runtime.	Assign Business Identifiers for Tracking Fields in Messages and Track Integration Instances in *Using Integrations in Oracle Integration 3*
8	Manage errors at the integration level, connection level, or specific integration instance level.	Manage Errors in *Using Integrations in Oracle Integration 3*

2
Oracle Utilities Adapter Concepts

The following sections describe Oracle Utilities Adapter capabilities in detail.

- Authentication Support
- Mapper Connectivity Properties Support
- Asynchronous Trigger Support in Orchestrated Integrations

Authentication Support

The following sections describe Oracle Utilities Adapter authentication capabilities in detail.

- About OAuth 2.0 Grants
- Use OAuth 2.0 Grants
- Authentication Types

About OAuth 2.0 Grants

The following sections describe Oracle Utilities Adapter authentication capabilities in detail.

This authentication scheme enables external clients to acquire a token that is also sent as part of the request sent to invoke Oracle Utilities application APIs.

The most important step for an application in the OAuth flow is how the application receives an access token (and optionally a refresh token). A grant type is the mechanism used to retrieve the token. OAuth defines several different access grant types that represent different authorization mechanisms.

Applications can request an access token to access protected endpoints in different ways, depending on the type of grant type specified in the Oracle Identity Cloud Service application. A grant is a credential representing the resource owner's authorization to access a protected resource.

The following sections discuss the various grant types and their pros/cons, along with instructions on how to configure the specific grant type.

There are several OAuth 2.0 grant types you can use in Oracle Integration with the Oracle Utilities Adapter. Review the following information to identify the grant type to use for your use case.

Grant Type	About the Grant Type	Use Cases and Risks
JWT user assertion (recommended)	A user assertion is a user token that contains identity information about the user. The user can either represent a human or a service integration account created for identifying a specific calling application. The user assertion is used directly as an authorization grant to obtain an access token. The client details are provided either as an authentication header in the request or as a client assertion. The user assertion grant is more secure than the resource owner password credentials grant because the user's credentials are never exposed. The user assertion workflow: • Is used with confidential clients. The OAuth clients are trusted to assert a user/ service integration account identity on behalf of the user/ service integration account. • The resource owner's credentials (Oracle Integration user) are never accessible to the client application. It just uses the assertion of the resource owner. • It isn't redirection-based. It takes a request only from the client application to the authorization server. The user is not redirected between interfaces to authorize the request. This user assertion grant works as follows: • The client requests an access token by providing a user assertion. The client details are provided either as an authentication header in the request or as a client assertion. • The OAuth service authenticates the client and, if valid, supplies an access token. The JWT user assertion characteristics are as follows:	This grant is used by applications that want to programmatically invoke integrations without any user intervention. The client application impersonates the user by sending the user assertion to Oracle Identity Cloud Service while requesting token access. An access token is returned in the user context. The user can either represent a human or a service integration account created for identifying a specific calling application. Oracle Integration recommends the use of this grant for acquiring an OAuth access token by the applications that must programmatically start the integration without any user intervention. **Risks** Carefully use this grant (only with first party/trusted clients) because it allows for trivial impersonation to more highly privileged accounts on services. **Usage** See Prerequisites for Creating a Connection.

Grant Type	About the Grant Type	Use Cases and Risks
	• Does not require the client to have knowledge of user credentials. • There is no browser-based end user interaction. • An access token is in the context of the end user. In this OAuth flow: • A user attempts to access a client application by sending a generated user assertion. • The client application requests an access token, and often a refresh token, by providing a user assertion or a third-party user assertion. • The Oracle Identity Cloud Service authorization server returns the access token to the client application. • The client application uses the access token in an API call to invoke the Utilities Application services.	

Grant Type	About the Grant Type	Use Cases and Risks
Resource owner password credential (ROPC)	The resource owner's password credentials (that is, the user name and password) can be used by the OAuth client directly as an authorization grant to obtain an access token. The resource owner password credentials grant type is suitable for cases where the resource owner has a trust relationship with the OAuth client. When using the resource owner password credentials grant, the user provides the credentials (user name and password) directly to the application. The application then uses the credentials to obtain an access token from the OAuth token service. The resource owner password credentials grant is a grant workflow where the client application, together with its client identifier and secret, sends the user name and password in exchange for an access token. Instead of the user having to log in and approve the authorization request in a web interface, the user can enter the user name and password in the client application user interface directly. This workflow has different security properties than other OAuth workflows. The primary difference is that the user's password is accessible to the application. This requires a strong trust of the application by the user. The resource owner password credentials grant has the following characteristics: • The client is required to have knowledge of user credentials. • Is not a browser-based end user interaction. • A refresh token is allowed. • An access token is in the context of the end user. In this OAuth flow: • The user requests the protected resource. • The client application requests the resource	This grant can be used by applications that want to programmatically invoke the integration without any user intervention. Use this grant only with trusted first-party clients that securely handle user credentials. Even though this grant type can be used by client applications to acquire an OAuth access token to use for sending the request to invoke an integration in a programmatic manner, Oracle Integration does *not* recommend the resource owner password credential grant because of the following risks: **Risks** • This grant type carries a higher risk than other grant types because it maintains the password anti-pattern this protocol seeks to avoid. The client can abuse the password or the password can unintentionally be disclosed to an attacker (for example, through log files or other records kept by the client). • The application can request a scope with complete access to user resources once it possesses the password credential. • Passwords expire. **Usage** See Prerequisites for Resource Owner Password Credentials.

Grant Type	About the Grant Type	Use Cases and Risks
	owner's user name and password. • The user logs in with their user name and password. • The client application exchanges those credentials for an access token, and often a refresh token, from the Oracle Identity Cloud Service authorization server. • The Oracle Identity Cloud Service authorization server returns the access token to the client application. • The client application uses the access token in an API call to invoke the integration.	

Grant Type	About the Grant Type	Use Cases and Risks
Client credentials	The client credentials grant is used when the OAuth client itself owns the data and doesn't need delegated access from a resource owner. The client credentials grant provides a specific grant flow in which the resource owner (that is, the user) is not involved. When using this grant, the client application requests an access token only with its own credentials (the identifier and secret) or an assertion from the token endpoint and uses the access token on behalf of the client application itself. The token endpoint does not issue a refresh token. This is because refresh tokens are not supported by the client credentials grant. When the access token expires, the client application must request a new access token. Client credentials have the following characteristics: • Are used by confidential OAuth clients. • The OAuth client application communicates with the service provider directly and not on behalf of a resource owner. • The flow is not redirection-based. • An access token is outside of the context of the end user. In this OAuth flow: • The user requests the protected resource. • The client application exchanges its credentials for an access token from the Oracle Identity Cloud Service authorization server. • The Oracle Identity Cloud Service authorization server validates the client credentials and returns the access token to the client application. • The client application uses the access token in an API call to invoke the integration.	A OAuth client that uses the client credentials grant must have credentials on the authorization server, which means the client must be a confidential client. With the client credentials grant, the integration server does not know the owner because the client does not need resource owner approval. This grant flow is best-suited for when a service provider wants to provide some API methods for use by the client application in general, instead of methods that apply to a certain resource owner. **Usage** See Prerequisites for OAuth Client Credentials

Use OAuth 2.0 Grants

To use an OAuth 2.0 grant type with the Oracle Utilities Adapter in Oracle Integration, you must perform the following prerequisites.

- Prerequisites for All Grants
- Prerequisites for Resource Owner Password Credentials
- Prerequisites for OAuth Client Credentials
- Prerequisites for JWT Assertions

Note:

Understand the following restrictions before performing OAuth 2.0 grants.

- Do *not* let external client applications use the system-created Oracle Identity Cloud Service application to authenticate against Oracle Integration endpoints.
- The scope of the client application is for accessing all deployed integrations in that service instance. There is *no* support for limiting access to a subset of integrations.
- To trigger the Oracle Integration flow with OAuth, the Oracle Utilities Adapter trigger connection should also use the OAuth Client Credentials security policy.
- The OUAF applications do not support the OAuth Client Credentials using JWT Client Assertion security policy or the OAuth using JWT User Assertion security policy for the message sender to trigger an integration. However, the Basic Authentication and OAuth Client Credentials security policies can still be used by the message sender to initiate a JWT-based connection on the trigger end of the integration.

Prerequisites for All Grants

Perform the following tasks for each grant type you use.

- Obtain the Oracle Identity Cloud Service URL.
  1. Go to the URL for your Oracle Utilities application. You are redirected to a URL such as:

     ```
     https://idcs-c2881.identity.myhost.example.com/ui/v1/signin
     ```

  2. Replace `/signin` with `/adminconsole` to access Oracle Identity Cloud Service. For example:

     ```
     https://idcs-c2881.identity.myhost.example.com/ui/v1/adminconsole
     ```

     You'll be prompted to sign in again to the Oracle Identity Cloud Service Console.
  3. Log in to the Oracle Identity Cloud Service Console with your identity domain administrator credentials.
- Check the Oracle Utilities application in Oracle Identity Cloud Service.

When an Oracle Utilities application instance is provisioned, an Oracle Identity Cloud Service application is created for that application instance. The application name is composed as follows:

```
product-domaintenantsuffixsequential_number
```

For example:

```
CCS-PRODC12345CMETERDATA0
```

```
CCS-PRODC12345FIELDSERVICE1
```

1. To request creation of a new OAuth client application, create a cloud operations service request and provide the following information:
   - Environment(s) where the OAuth client application is needed (for example, PROD, TEST01, or DEV).
   - Client name suffix: Use a distinct name that may suggest the functional purpose of the integration.
   - Provide a meaningful description of the integration point.
   - Client type (trusted or confidential) and client certificate: The integration requirements may call for a trusted client and the external application may also supply its own certificate. Otherwise, Oracle Identity Cloud Service creates a trusted client with its internal native certificate.
   - OAuth flow for your intended integration: Client credentials is currently supported.
   - Scope: You can define the OAuth client application with access to either REST or SOAP APIs or both REST and SOAP APIs.

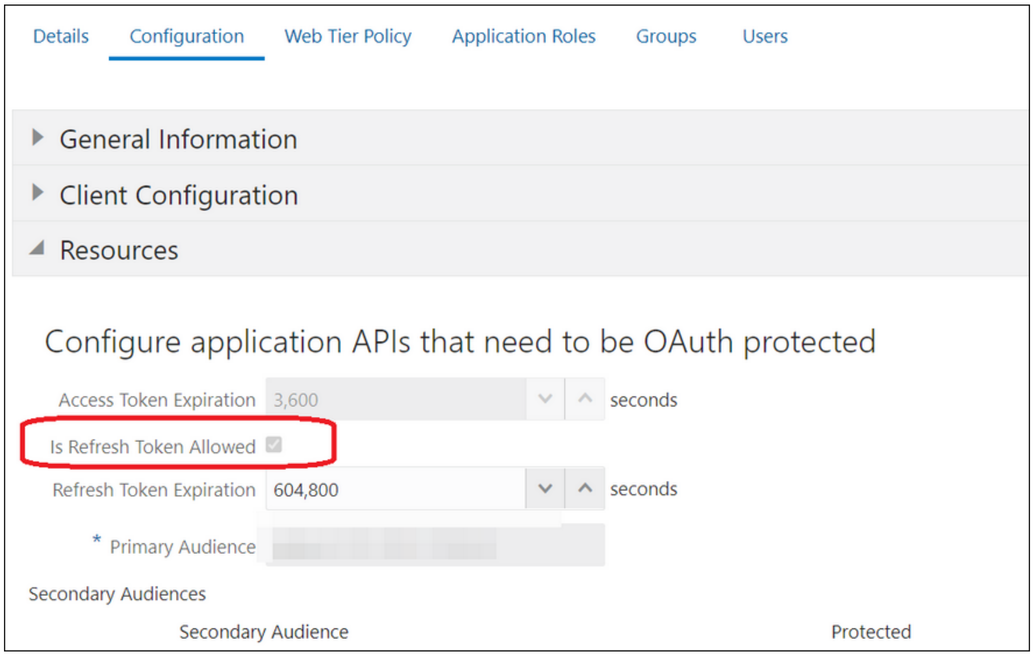
   The Oracle Utilities Cloud Operations team creates the OAuth client using the input provided in the service request.

   1. Log in to Oracle Identity Cloud Service to get your application.
   2. Go to Oracle Cloud Services and find the application with the above name to access the application.

Prerequisites for Resource Owner Password Credentials

Perform the following tasks.

- Validate the Oracle Integration application and user roles:
  1. Go to **Configuration**, and then **Client Configuration** of the Oracle Identity Cloud Service application.
  2. Verify that **Resource Owner** and **Refresh Token** for **Allowed Grant Types** are enabled.
  3. Go to **Configuration**, and then **Resources** of the Oracle Identity Cloud Service application.
  4. Verify that the **Is Refresh Token Allowed** option is enabled.



Details | Configuration | Web Tier Policy | Application Roles | Groups | Users

General Information

Client Configuration

Resources

Configure application APIs that need to be OAuth protected

Access Token Expiration 3,600 seconds

Is Refresh Token Allowed ✓

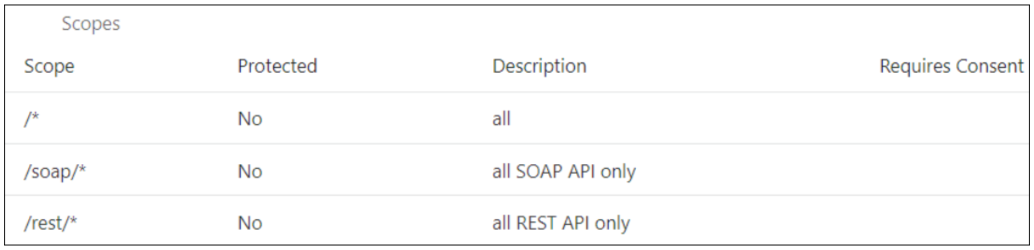
Refresh Token Expiration 604,800 seconds

* Primary Audience [redacted]

Secondary Audiences

Secondary Audience	Protected

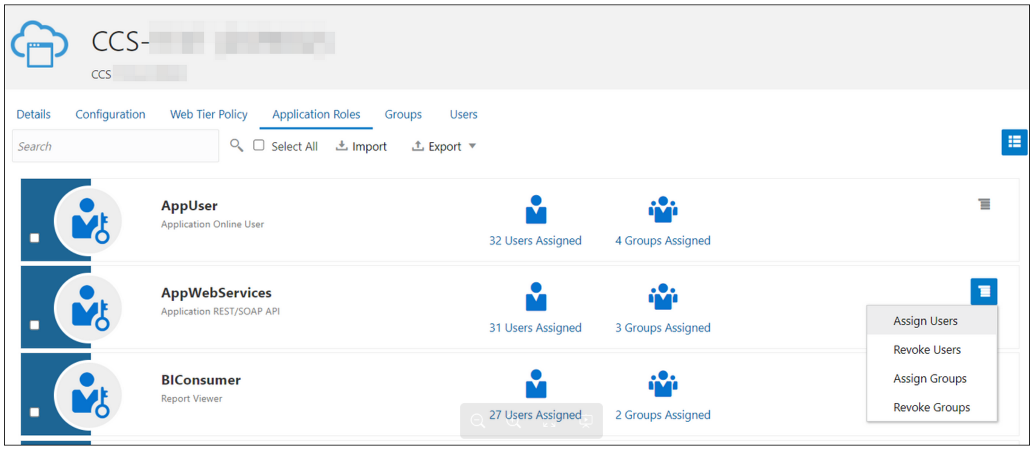
The scope with access to either REST or SOAP APIs or both REST and SOAP APIs is provided.



Scopes

Scope	Protected	Description	Requires Consent
/*	No	all	
/soap/*	No	all SOAP API only	
/rest/*	No	all REST API only	

5. Add the appropriate user(s) to the various Oracle Application roles. For standard/ production configurations, use the ServiceUser role. (See Oracle Integration Service Roles in *Provisioning and Administering Oracle Integration 3*.)
6. To assign the user, go to the **Application Roles** section of the application and assign the user for **AppWebServices**.



CCS-[redacted]

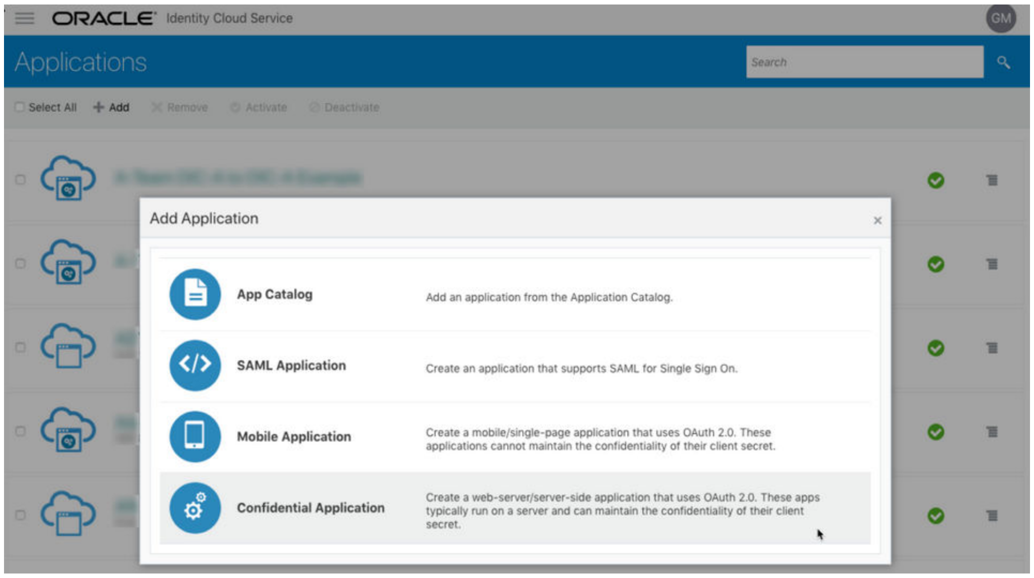
CCS [redacted]

Details | Configuration | Web Tier Policy | Application Roles | Groups | Users

Search | Select All | Import | Export

AppUser Application Online User	32 Users Assigned	4 Groups Assigned
AppWebServices Application REST/SOAP API	31 Users Assigned	3 Groups Assigned
BIConsumer Report Viewer	27 Users Assigned	2 Groups Assigned

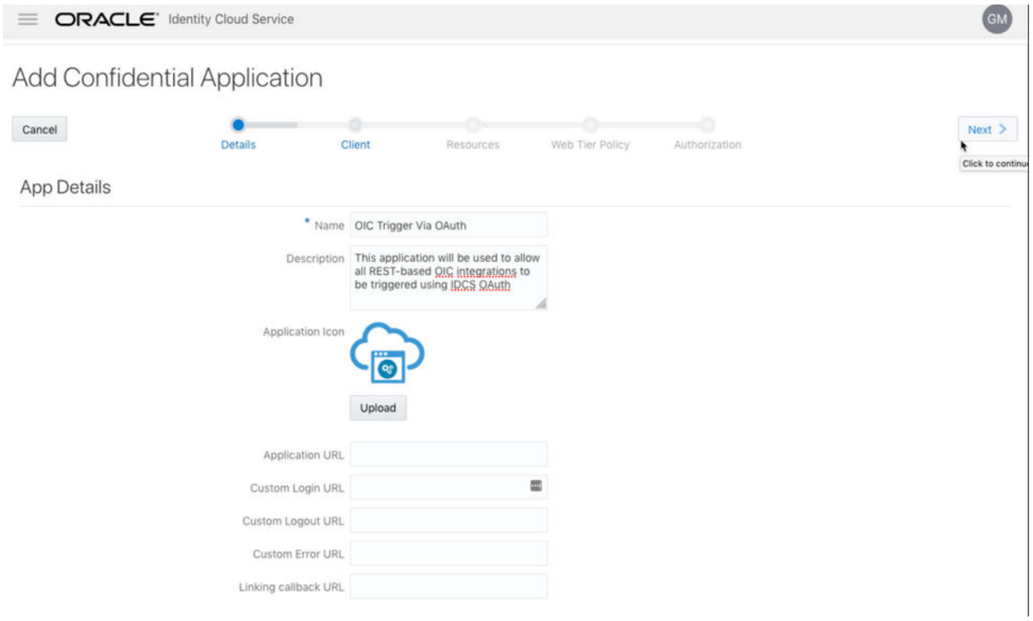
- Configure the client application:
  1. In the Oracle Identity Cloud Service Console, go to the **Applications** section to create a new application that allows you to invoke an Oracle Utilities application with an OAuth Utilities Connection. Add this application as a confidential application.
  2. Click **Add**.
  3. Select **Confidential Application**.



Add Application

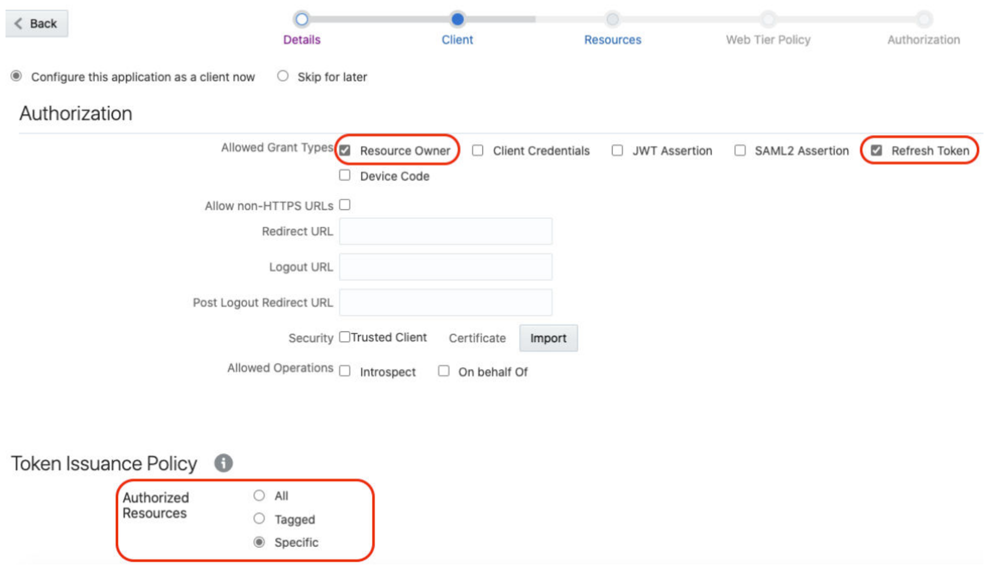
App Catalog	Add an application from the Application Catalog.
SAML Application	Create an application that supports SAML for Single Sign On.
Mobile Application	Create a mobile/single-page application that uses OAuth 2.0. These applications cannot maintain the confidentiality of their client secret.
Confidential Application	Create a web-server/server-side application that uses OAuth 2.0. These apps typically run on a server and can maintain the confidentiality of their client secret.

  4. Complete the **Details** page, and go to the **Client** page.



* Name	OIC Trigger Via OAuth
Description	This application will be used to allow all REST-based OIC integrations to be triggered using IDCS OAuth
Application Icon	
Application URL	
Custom Login URL	
Custom Logout URL	
Custom Error URL	
Linking callback URL	

  5. On the **Client** page, select **Configure this application as a client now** and add the following.
     a. Select **Resource Owner** and **Refresh Token** for **Allowed Grant Types**.
     b. Select **Specific** in the **Authorized Resources** section.



Configure this application as a client now [✓] Skip for later

Authorization

Allowed Grant Types: [✓] Resource Owner, [ ] Client Credentials, [ ] JWT Assertion, [ ] SAML2 Assertion, [✓] Refresh Token, [ ] Device Code

Allow non-HTTPS URLs [ ]

Redirect URL

Logout URL

Post Logout Redirect URL

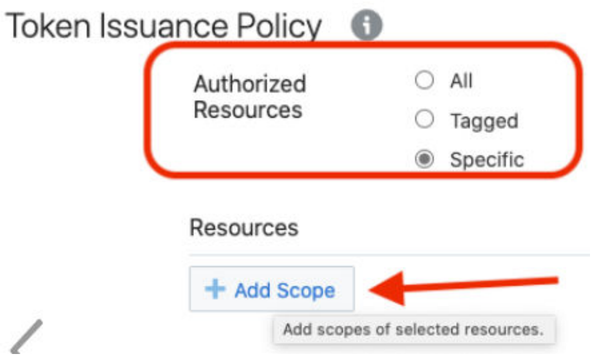
Security [ ] Trusted Client Certificate Import

Allowed Operations [ ] Introspect [ ] On behalf Of

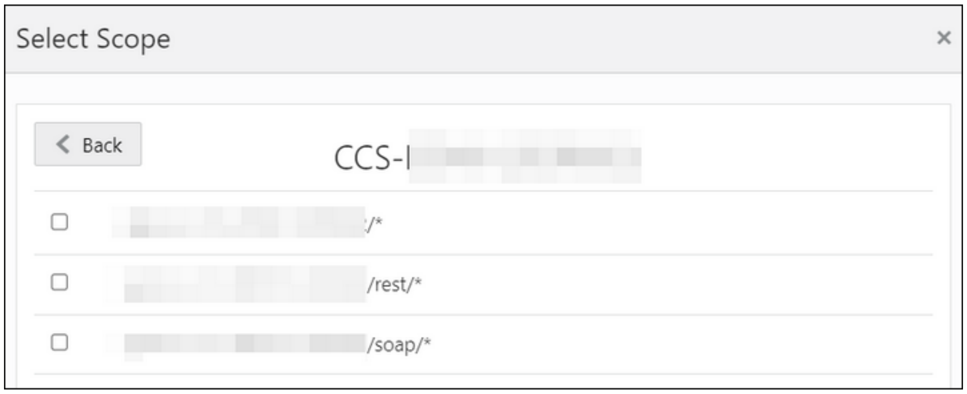
Token Issuance Policy

Authorized Resources: All / Tagged / [✓ Specific]

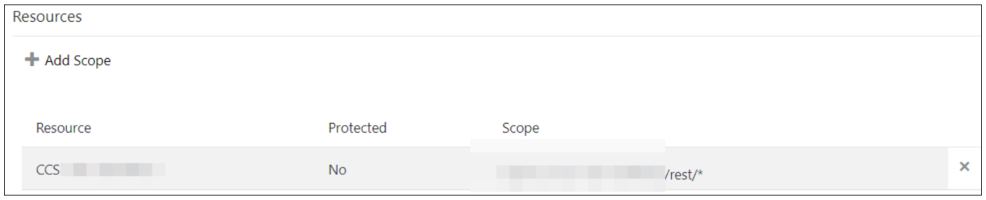
c. Click **Add Scope** under the **Resources** section.



d. Find the Oracle Utilities application.



e. Add the scope containing access to either REST (**/rest/***) or SOAP(**/soap/***) APIs or both REST and SOAP APIs (**/***), and click **>**.



Resource	Protected	Scope
CCS [illegible]	No	[illegible]/rest/*

  f. Save your changes.

6. Click through the remaining wizard pages without making changes and save the application.
7. Activate the application for use.

- Validate the client application:

  1. To fetch the access client, make a request to Oracle Identity Cloud Service with the user name and password in the payload.

     > **Note:**
     >
     > Add `offline_access` in the scope to fetch the request refresh token as part of the response.

     ```
     ##Syntax
     curl -i -H 'Authorization: Basic <base64Encoded_clientid:secret>' -H
     'Content-Type: application/x-www-form-urlencoded;charset=UTF-8' --
     request POST https://<IDCS-Service-Instance>.identity.oraclecloud.com/
     oauth2/v1/token -d 'grant_type=password&username=<user-
     name>&password=<password>&scope=<App_Scope>%20offline_access'

     ###where
     #### <base64-clientid-secret> - Base 64 encode clientId:ClientSecret
     #### <username> - user for token needs to be issued (must be in
     serviceuser role).
     #### <password> - password for above user
     #### <app_scope> - Scope added while creating application in client
     configuration section
     ##Example
     curl -i -H 'Authorization: Basic OGQyM...ZDA0Mjcz' -H 'Content-Type:
     application/x-www-form-urlencoded;charset=UTF-8' --request POST https://
     <idcs_host>/oauth2/v1/token -d
     'grant_type=password&username=sampleUser&password=SamplePassword&scope=h
     ttps://<Resource_APP_Audience>/rest/*%20offline_access'
     ```

  2. Capture the `access_token` and `refresh_token` from the response.

     ```
     {
         "access_token": "eyJ4NXQjG...dfsdfsFgets2ed",
         "token_type": "Bearer",
         "expires_in": 3600,
         "refresh_token": "AQAgY2MzNjVlOTVhOTRh...vM5S0MkrFSpzc="
     }
     ```

3. Use the `access_token` in the authorization header to invoke the Oracle Utilities application endpoint.

   ```
   curl --location --request GET 'https://
   <Utilities_Application_API_ENDPOINT>' \
   --header 'Authorization: Bearer eyJ4NXQjG...dfsdfsFgets2ed'
   ```

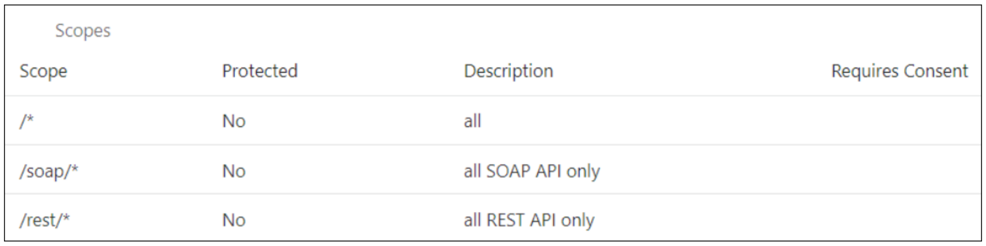
4. To update the access token, use the refresh token and make a request to Oracle Identity Cloud Service.
5. Capture the `access_token` and `refresh_token` from the response for further use.

   ```
   curl -i -H 'Authorization: Basic <base64-clientid-secret>' -H 'Content-
   Type: application/x-www-form-urlencoded;charset=UTF-8' --request POST
   https://<IDCS-Service-Instance>.identity.oraclecloud.com/oauth2/v1/
   token  -d 'grant_type=refresh_token&refresh_token=<refresh_token>'

   ##Example
   curl -i -H 'Authorization: Basic OGQyM...ZDA0Mjcz' -H 'Content-Type:
   application/x-www-form-urlencoded;charset=UTF-8' --request POST https://
   <IDCS-Service-Instance>.identity.oraclecloud.com/oauth2/v1/token  -d
   'grant_type=refresh_token&refresh_token=AQAgY2MzNjVlOTVhOTRh...vM5S0MkrF
   Spzc='
   ```

Prerequisites for OAuth Client Credentials

Oracle Identity Cloud Service (IDCS) configuration requirements for OAuth Client Credential inbound and outbound requests in integrations using the Oracle Utilities Adapter are described below.

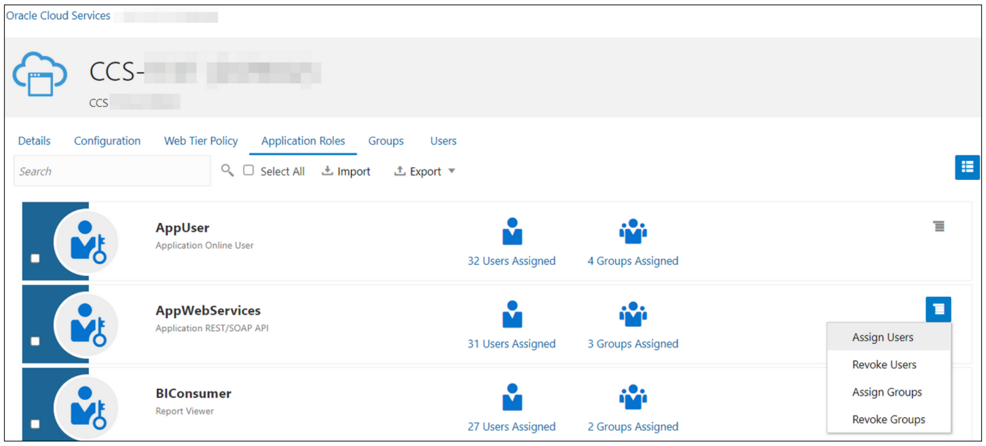
- **Inbound to Oracle Integration**:
  For IDCS configurations to enable OAuth Client Credentials in the inbound direction to Oracle Integration, see Prerequisites for Client Credentials in *Using the REST Adapter with Oracle Integration 3*.
- **Outbound from Oracle Integration to Oracle Utilities Applications:**
  Perform the following tasks.
  – Validate the Oracle Utilities application and user roles:
    1. Go to **Configuration**, and then **Client Configuration** of the Oracle Identity Cloud Service application.
    2. Verify that **Client Credentials** for **Allowed Grant Types** is enabled.
    3. Go to **Configuration**, and then **Resources** of the Oracle Identity Cloud Service application. The scope with access to either REST or SOAP APIs or both REST and SOAP APIs is provided.



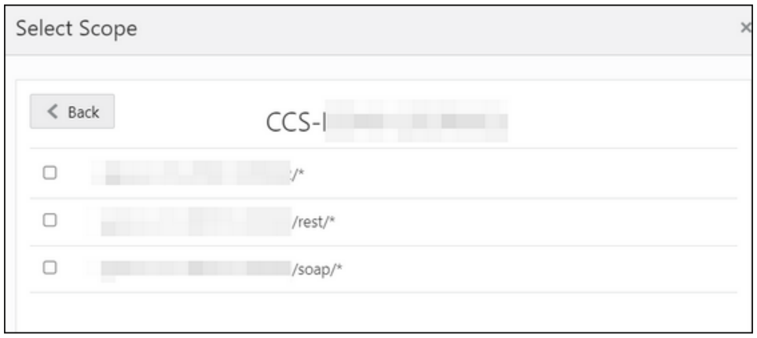
Scopes

Scope	Protected	Description	Requires Consent
/*	No	all	
/soap/*	No	all SOAP API only	
/rest/*	No	all REST API only	

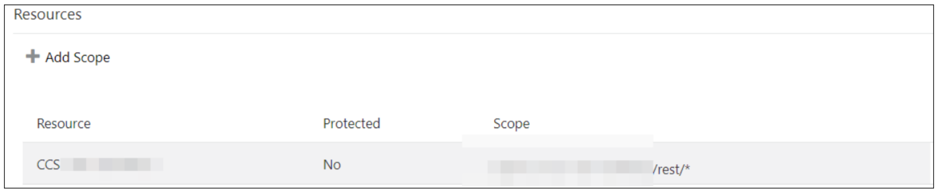
4. Add the appropriate user(s) to the various Oracle Utilities application roles. For standard/production configurations, use the ServiceUser role. (See Oracle Integration Service Roles in *Provisioning and Administering Oracle Integration 3*.)
5. To assign the user, go to the **Application Roles** section of the application under **AppWebServices**.



– Configure the client application:
1. In the Oracle Identity Cloud Service Console, go to the **Applications** section to create a new application that allows you to invoke an Oracle Utilities application API with an OAuth Utilities Connection. The application is added as a confidential application.
2. Complete the **Details** section, and go to the **Client** section.
3. On the **Client** page, select **Configure this application as a client now**, and complete the following:
   a. Select **Client Credentials** from the **Allowed Grant Types** list.
   b. Select **Specific** in the **Authorized Resources** area of the **Token Issuance Policy** section.
   c. Click **Add Scope** under the **Resources** section.
   d. Find the Oracle Utilities application.



   e. Add the scope containing access to either REST or SOAP APIs or both REST and SOAP APIs, and click **>**.



Resource	Protected	Scope
CCS [redacted]	No	[redacted]/rest/*

f. Save your changes.

4. Click through the remaining wizard pages without making changes and save the application.

5. Activate the application for use.

The next step is to create an application user in the appropriate Oracle Utilities Cloud Service (such as Oracle Utilities Meter Solution Cloud Service). Access the appropriate Oracle Utilities Cloud Service application, and navigate to the User portal.

1. Create a new user corresponding to the OAuth client created above:
   a. Search for `Add User`.
   b. Enter the OAuth client ID as the user's login ID.
   c. Provide values of your choice for the mandatory fields, such as **USER**, **LAST NAME**, and **FIRST NAME**.
   d. Select **USER ENBLE** as **Enable**.
   e. Assign **User Groups** that provide the integration with access to the appropriate functionality (for example, the **ALL_SERVICES** user group providing a future expiration date).
   f. Assign **To Do Roles** (for example, the **F1_DFLT** system default role) and appropriate **Access Security** (for example, **System Default**) with an expiration date to the created user.
   g. Save the user.

   The OAuth client credentials are now ready to use.

– Validate the client application.

1. Fetch the access client to make a request to Oracle Identity Cloud Service with the client ID and client secret of the client in the payload.

```
##Syntax
curl -i -H 'Authorization: Basic <base64Encoded_clientid:secret>' -
H 'Content-Type: application/x-www-form-urlencoded;charset=UTF-8' --
request POST https://<IDCS-Service-
Instance>.identity.oraclecloud.com/oauth2/v1/token -d
'grant_type=client_credentials&scope=<App_Scope>'

###where
#### <base64-clientid-secret> - Base 64 encode clientId:ClientSecret
#### <app_scope> - Scope added while creating application in client
configuration section
##Example
curl -i -H 'Authorization: Basic OGQyM...ZDA0Mjcz' -H 'Content-
Type: application/x-www-form-urlencoded;charset=UTF-8' --request
POST https://<idcs_host>/oauth2/v1/token -d
```

```
'grant_type=client_credentials&scope=https://
<Resource_APP_Audience>/rest/*'
```

2. Capture the `access_token` from the response.

   ```
   {
       "access_token": "eyJ4NXQjG...dfsdfsFgets2ed",
       "token_type": "Bearer",
       "expires_in": 3600
   }
   ```

3. Use the `access_token` in the authorization header to invoke the Oracle Utilities Application APIs.

   ```
   curl --location --request GET 'https://
   <Utilities_Application_API_ENDPOINT>' \
   --header 'Authorization: Bearer eyJ4NXQjG...dfsdfsFgets2ed'
   ```

4. To update the access token when it is expired, make the same request to Oracle Identity Cloud Service.

Prerequisites for JWT Assertions

The following section describes Oracle Identity Cloud Service (IDCS) configuration requirements for JWT user/client assertion inbound and outbound requests in integrations using the Oracle Utilities Adapter.

- Inbound to Oracle Integration
  For IDCS configurations to enable JWT user/client assertions in the inbound direction to Oracle Integration, see Prerequisites for JWT User Assertion in *Using the REST Adapter with Oracle Integration 3*.
- Outbound from Oracle Integration to Oracle Utilities Applications

Perform the following tasks:

- Generate the key
- Configure the client application
- Add a certificate as a trusted partner
- Generate the JWT user assertion
- Generate the JWT client assertion
- Validate the client application

- **Generate the key**
  You must first generate the key to import when you configure the client application for the JWT user assertion.
  1. Generate the self-signed key pair.

     ```
     keytool -genkey -keyalg RSA -alias <your_alias> -keystore
     <keystore_file> -storepass <password> -validity 365 -keysize 2048

     ##example
     keytool -genkey -keyalg RSA -alias assert -keystore sampleKeystore.jks -
     storepass samplePasswd -validity 365 -keysize 2048
     ```

2. Export the public key for signing the JWT assertion.

   ```
   keytool -exportcert -alias <your_alias> -file <filename> -keystore
   <keystore_file> -storepass <password>

   ##example
   keytool -exportcert -alias assert -file assert.cer -keystore
   sampleKeystore.jks -storepass samplePasswd

   ## This should show a success message e.g. Certificate stored in file
   <assert.cer>
   ```

3. Convert the keystore to P12 format.

   ```
   keytool -importkeystore -srckeystore <filename> -srcstorepass
   <password> -srckeypass <password> -srcalias <your_alias> -destalias
   <your_alias> -destkeystore <destFileName> -deststoretype PKCS12 -
   deststorepass <password> -destkeypass <password>

   ##example
   keytool -importkeystore -srckeystore sampleKeystore.jks -srcstorepass
   samplePasswd -srckeypass samplePasswd -srcalias assert -destalias
   assert -destkeystore assert.p12 -deststoretype PKCS12 -deststorepass
   samplePasswd -destkeypass samplePasswd

   ## This should show a success message e.g. Importing keystore
   sampleKeystore.jks to assert.p12...
   ```

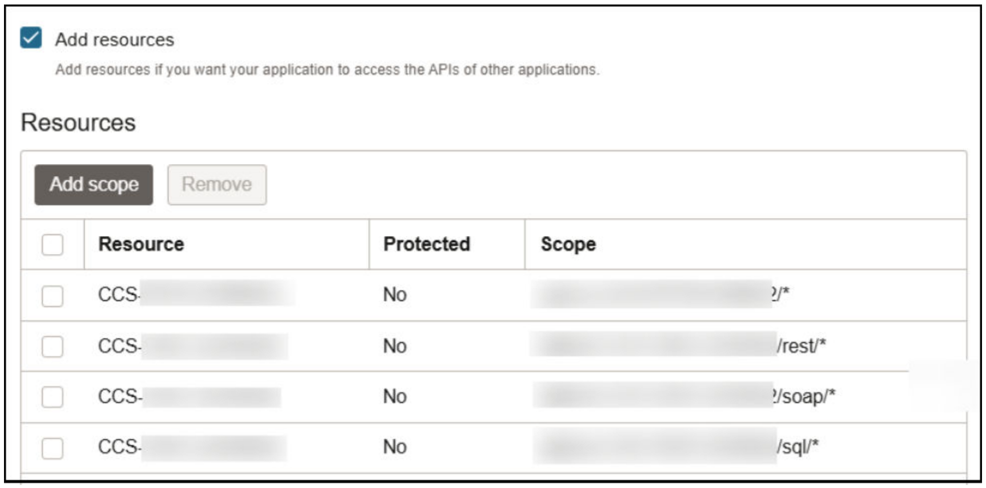
4. Export the private key from the P12 keystore.

   ```
   openssl pkcs12 -in <destFileName> -nodes -nocerts -out <pem_file>

   ##example
   openssl pkcs12 -in assert.p12 -nodes -nocerts -out private_key.pem

   ## This should show a success message: MAC verified OK
   ```

- **Configure the client application**
  In the Oracle Cloud Infrastructure Console, go to the **Integrated applications** section to create a new application that allows you to invoke an Oracle Utilities application API with an JWT Utilities connection.

  1. Click **Add application**.
  2. Select **Confidential Application**, and click **Launch workflow**.
  3. Enter a name in the **Application details** section. The remaining fields on this page are optional and can be ignored.
  4. Click **Next**.
  5. In the **Client configuration** section, select **Configure this application as a client now**, and complete the following:
     a. For JWT assertions, select **JWT assertion** and **Refresh token** in the **Allowed grant types** section.
     b. Leave the **Redirect URL**, **Post-logout redirect URL**, and **Logout URL** fields blank.

c. In the **Client type** section, select **Trusted**.
d. Upload the public certificate created in the Generate the key section.
e. Provide the alias name. Remember this alias name because it is used when creating assertions. This action adds the certificate as a trusted partner.
f. Bypass several fields and scroll down to the **Token issuance policy** section.
g. Select **Specific** in the **Authorized resources** section.
h. Click the **Add Resources** check box.
i. Click **Add scope**.
j. Use the search facility to find the Oracle Utilities application.

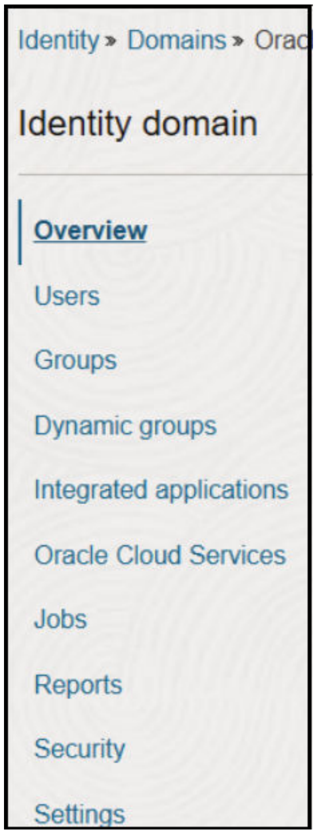

Add resources
Add resources if you want your application to access the APIs of other applications.
Resources
Add scope
Remove

	Resource	Protected	Scope
☐	CCS-	No	/*
☐	CCS-	No	/rest/*
☐	CCS-	No	/soap/*
☐	CCS-	No	/sql/*

k. Add the scope containing access to either REST or SOAP APIs or both REST and SOAP APIs. The scopes are displayed in the **Resources** section.
l. Click through the remaining wizard pages without making changes and save the application.

6. Activate the application for use.
7. In the **General Information** section, note the client ID and client secret values. These values are required for the third-party application that is communicating with the identity domain.

- **Add a certificate as a trusted partner**
  In addition to importing the signing certificate into the client application, you are also required to include the certificate as a trusted partner certificate.
  1. In the navigation pane, click **Settings**.



2. Click **Trusted partner certificates**.
3. Click **Import certificate** to upload the public certificate created in the Generate the key section.

- **Generate the JWT user assertion**
  Generate the JWT user assertion using the generated private key and simple Java code.

Note:

You can use the https://github.com/jwtk/jjwt library to generate the user assertion. There are many libraries listed at https://jwt.io/ for multiple technologies.

```
Sample:
header:
{
"alg": "RS256",
"typ": "JWT",
"kid": "assert"
}

payload:
{
"sub": "utilitiesApplicationUser",
"jti": "8c7df446-bfae-40be-be09-0ab55c655436",
"iat": 1589889699,
"exp": 1589909699,
"iss": "d702f5b31ee645ecbc49d05983aaee54",
"aud": "https://identity.oraclecloud.com/"
}
```

Where:

- `sub` specifies the user name for whom user assertion is generated.
- `jti` is a unique identifier.
- `iat` is issued (epoch seconds).
- `exp` is the token expiry (epoch seconds).
- `iss` is the client ID aud must include the identity domain audience `https://identity.oracle.com/`. The signing algorithm must be RS256.
- `kid` specifies the key to use to verify the signature. Therefore, it must match with the uploaded certificate alias.

* **Generate the JWT client assertion**
  Similar to JWT user assertion, you can generate the client assertion using the generated private key and simple Java code.

  > **Note:**
  >
  > You can use the https://github.com/jwtk/jjwt library to generate the user assertion. There are many libraries listed at https://jwt.io/ for multiple technologies.

```
Sample:
header:
{
"alg": "RS256",
"typ": "JWT",
"kid": "assert"
}

payload:
{
"sub": "d702f5b31ee645ecbc49d05983aaee54",
"jti": "8c7df446-bfae-40be-be09-0ab55c655436",
"iat": 1589889699,
"exp": 1589909699,
"iss": "d702f5b31ee645ecbc49d05983aaee54",
"aud": "https://identity.oraclecloud.com/"
}
```

Where:

- `sub` specifies the client ID of the client application.
- `jti` is a unique identifier.
- `iat` is issued (epoch seconds).
- `exp` is the token expiry (epoch seconds).

- `iss` is the client ID aud must include the identity domain audience https://identity.oracle.com/. The signing algorithm must be RS256.
- `kid` specifies the key to use to verify the signature. Therefore, it must match with the uploaded certificate alias.

> **Note:**
>
> In user assertions, you add `sub` as the user name. In client assertions, you add the client ID as `sub`.

- **Validate the client application**

1. Once you generate the JWT user assertion, generate the access token as follows.

```
##Syntax
curl -i -H 'Authorization: Basic <base64Encoded clientid:secret>' -H
'Content-Type: application/x-www-form-urlencoded;charset=UTF-8' --
request POST https://
<Identity_Domain_Service_Instance>.identity.oraclecloud.com/oauth2/v1/
token -d 'grant_type=urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Ajwt-
bearer&assertion=<user assertion>&scope=<app_scope>'

###where
#### grant type - urn:ietf:params:oauth:grant-type:jwt-bearer
#### <base64-clientid-secret> - Base 64 encode clientId:ClientSecret
#### <user assertion> - User assertion generated
#### <app scope> - Scope added while creating application in client
configuration section
```

2. Capture the `access_token` from the response.

```
{ "access_token": "eyJ4NXQjG...dfsdfsFgets2ed", "token_type": "Bearer",
"expires_in": 3600 }
```

3. Use the `access_token` in the authorization header to invoke the Oracle Utilities Application APIs.

```
curl --location --request GET 'https://
<Utilities_Application_API_ENDPOINT>' \ --header 'Authorization: Bearer
eyJ4NXQjG...dfsdfsFgets2ed'
```

4. To update the access token when it expires, make the same request to Oracle Identity Cloud Service.

- **Add new security policies**
  - OAuth Client Credentials using JWT Client Assertion
  - OAuth using JWT User Assertion

Authentication Types

The Oracle Utilities Adapter supports the following types of authentication:

- Username password token

- Basic Authentication
- OAuth Client Credentials (two-legged flow)
- OAuth Resource Owner Password Credentials (two-legged flow)
- OAuth using JWT User Assertion
- OAuth Client Credentials using JWT Client Assertion

Additional information about these security policies is provided. See Configure Connection Security.

Mapper Connectivity Properties Support

The following section describes Oracle Utilities Adapter mapper connectivity property capabilities in more detail.

> **Note:**
>
> This support is only available for non-OUAF (NMS) applications.

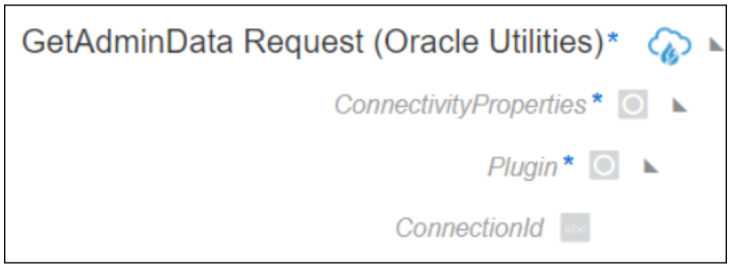
- Set the Oracle Utilities Adapter Connectivity Properties in the Mapper
- Support for Dynamic REST Endpoints
- Limitation of Connection Properties with OAuth Client Credentials

Set the Oracle Utilities Adapter Connectivity Properties in the Mapper

You can set Oracle Utilities Adapter connectivity properties in the mapper to propagate additional information to the target endpoint during runtime.

- Connectivity properties (invoke request)
  You can set the following property for invoke requests in the mapper.

Property	Description
ConnectionId	The connection identifier value is passed to the mapping to fetch the respective connection details using the lookup for invoking a dynamic non-OUAF application during runtime.



Support for Dynamic REST Endpoints

The Oracle Utilities Adapter enables you to dynamically change the (invoke) outbound endpoint configuration. This feature is useful in the following scenarios:

- A REST endpoint must be invoked dynamically at runtime.
- An endpoint is not known at design time.

To change the endpoint configuration at runtime, you must map the properties under **ConnectivityProperties** in the mapper.

For example, the following steps describe how to configure an integration to invoke a REST endpoint determined at runtime:

1. Create multiple connections with the Oracle Utilities Adapter and configure each environment detail for non-OUAF applications that can be invoked during the runtime.
2. Create a lookup having a key-value pair and configure the connection identifier values of the above created connections as identifier values.
3. Create an integration adding any connection of the non-OUAF application on the invoke.
4. In the target pane of the mapper, expand **Plugin** under **ConnectivityProperties**. These elements are made available automatically through a static schema that is added to the user-provided schema.
5. Using the source schema in the source pane, create a mapping to **ConnectionId** in the target pane. Alternatively, you can also provide a static mapping. The Oracle Utilities Adapter uses the **ConnectionId** provided by this mapping to determine the connection identifier configured in the lookup and fetch the connection details corresponding to this connection identifier. The respective REST endpoint is determined to which to route this request and the credentials to authenticate the request.
6. Activate and invoke the integration. The Oracle Utilities Adapter now invokes the endpoint URI determined at runtime.

Note:

With connection resources shared across projects, you must set the connection identifier as a **Public** connection in the lookup for the mapper instead of as a **Local** connection. To understand public and local connections in projects, see Add and Share a Connection Across a Project in *Using Integrations in Oracle Integration 3*.

Limitation of Connection Properties with OAuth Client Credentials

- You cannot use multiple connectivity agent setup with the same connectivity agent group for OAuth client credentials. This action can cause runtime failure.
- The enterprise application and the catalog should be running on same server host. Otherwise, this can cause runtime failure.

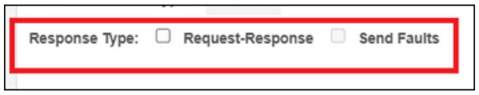
Asynchronous Trigger Support in Orchestrated Integrations

The web services you create have a request-response message exchange pattern (MEP) by default. In a request-response MEP in which the web service client invokes a method of the web service, the web service returns a response to the request.

The Oracle Utilities Adapter allows you to configure a web service to use a one-way message exchange pattern in which the web service client only sends a request message, but does not receive a response from the web service. This is supported in the trigger (inbound) direction for both REST and SOAP endpoints.

In the trigger direction, you must configure the Adapter Endpoint Configuration Wizard as follows:

1. Create an app-driven orchestrated integration.
2. Add an Oracle Utilities Adapter as a trigger connection in the integration. The Adapter Endpoint Configuration Wizard appears.
3. Configure the Basic Info page details.
4. Select the **One-way service** on the Request page.
5. Ensure that you configure the page as follows on the Response page. This ensures that the integration runs asynchronously.
   - The **Response Type** is enabled as **Request-Response** by default. Disable the **Request-Response** checkout for one-way support.
   - Once you disable the **Request-Response** checkbox, the Response page is not editable except for the **Request-Response** checkbox. No fault is accepted for one-way support. Therefore, the **Send Faults** selection is disabled.

   
   Response Type: ☐ Request-Response ☐ Send Faults

6. Click **Next** to access the Summary Page, then click **Done**.

3

Create an Oracle Utilities Adapter Connection

A connection is based on an adapter. You define connections to the specific cloud applications that you want to integrate.

Topics:

Prerequisites for Creating a Connection

Before you set up the Oracle Utilities Adapter:

- Upload a trusted public certificate (if required). Typically, the certificate is included with Oracle Integration. See Upload a Certificate to Connect with External Services.
- Make sure the Oracle Utilities server is running and accessible.
- Know the host name address and port number of the Oracle Utilities server.
- Know the user name and password used to access the Oracle Utilities server.
- Gather the required credentials and access token URI if using OAuth.
- Download the connectivity agent. See Download and Run the Connectivity Agent Installer in *Using Integrations in Oracle Integration 3*.
- Upload the required Oracle Identity Cloud Service certificate to the connectivity agent when using the OAuth Client Credentials security policy.

JWT Assertions Outbound Use

Perform the following prerequisites to use JWT assertions. Refer to the Identity Domain Cloud Services documentation to understand about the Assertions. See Client/User JWT Assertion.

- Manually create a signing key for upload on the Certificates page. See Upload a Certificate to Connect with External Services.
  The service provider typically provides instructions on how to generate the signing keys and the format. For an example, see Required Keys and OCIDs.
- Create the JWT header and JWT payload JSON files. You upload both files on the Connections page when configuring the Oracle Utilities Adapter to support JWT assertions. For example:

JWT Header JSON File Example	JWT Payload JSON File Example

`{`
`"alg": "RS256",`
`"typ": "JWT",`
`"kid": "assert"`
`}`
 	
Where:
– `alg`: Algorithm. Identifies the specific type of JWT signing algorithm. This is a required header for the JWT assertion. Oracle Identity Cloud Service currently supports RS256.
– `typ`: Type. Identifies the type of assertion, which is always JWT.
– `kid`: Key identifier. Used to identify the trusted third-party certificate to validate the assertion signature. The `x5t` or `kid` claim must be present in the JWT assertion header
	`{`
	`"sub": "utilitiesApplicationUser",`
	`"jti": "8c7df446-bfae-40be-be09-0ab55c655436",`
	`"iat": 1589889699,`
	`"exp": 1589909699,`
	`"iss": "d702f5b31ee645ecbc49d05983aaee54",`
	`"aud": "https://identity.oraclecloud.com/"`
	`}`
	
	Where:
	– JWT issuer (`iss`): A unique identifier for the entity that issued the assertion. This is typically the entity that holds the key material used to sign or integrity-protect the assertion. Examples of issuers are OAuth clients (when assertions are self-issued) and third-party security token services. If the assertion is self-issued, the issuer value is the client identifier (`client_id`). If the assertion was issued by a security token service (STS), the issuer must identify the STS in a manner recognized by the authorization server. The assertion must contain an issuer.
	– JWT subject (`sub`): The subject typically identifies an authorized accessor for which the access token is being requested (that is, the resource owner or an authorized delegate). In some cases, this may be a pseudo-anonymous identifier or other value denoting an anonymous user. For client assertions, the client ID value must be the Oracle Identity Cloud Service application `name` attribute. For user assertions, the claim value must be the user name.
	– JWT audience (`aud`): A value that identifies the party or parties to process the assertion. The assertion must contain an audience that identifies the authorization server as the intended audience. The authorization server must reject any assertion that does not contain its own identity as the intended audience (in this case, for Oracle Identity Cloud Service, `https://identity.oraclecloud.com/`).
	– Expires at (`exp`): The time (UNIX epoch time) at which the JWT assertion expires. This is a required claim for the assertion.
	– Issued at (`iat`): The date when the assertion was issued.
	– JWT ID (`jti`): The JWT ID claim is a unique identifier for the JWT.

> **Note:**
>
> Carefully review the JWT documentation of your service provider and ensure that you follow all required guidelines to correctly create the header and payload files. The content of each file varies from one service provider to another.

Create a Connection

Before you can build an integration, you must create the connections to the applications with which you want to share data.

> **Note:**
>
> You can also create a connection in the integration canvas. See why working with projects is preferred.

To create a connection in Oracle Integration:

1. Decide where to start:
   - Work in a project (see why working with projects is preferred).
     a. In the navigation pane, click **Projects**.
     b. Select the project name.
     c. Click **Integrations** ⊞.
     d. In the **Connections** section, click **Add** if no connections currently exist or **+** if connections already exist. The Create connection panel opens.
   - Work outside a project.
     a. In the navigation pane, click **Design**, then **Connections**.
     b. Click **Create**. The Create connection panel opens.
2. Select the adapter to use for this connection. To find the adapter, scroll through the list, or enter a partial or full name in the **Search** field.
3. Enter the information that describes this connection.

Element	Description
Name	Enter a meaningful name to help others find your connection when they begin to create their own integrations.
Identifier	Automatically displays the name in capital letters that you entered in the **Name** field. If you modify the identifier name, don't include blank spaces (for example, `SALES OPPORTUNITY`).

Element	Description
Role	Select the role (direction) in which to use this connection. **Note**: *Only* the roles supported by the adapter you selected are displayed for selection. Some adapters support all role combinations (trigger, invoke, or trigger and invoke). Other adapters support fewer role combinations. When you select a role, only the connection properties and security policies appropriate to that role are displayed on the Connections page. If you select an adapter that supports both invoke and trigger, but select only one of those roles, you'll get an error when you try to drag the adapter into the section you didn't select. For example, assume you configure a connection for the Oracle Service Cloud (RightNow) Adapter as only an **invoke**. Dragging the adapter to a **trigger** section in the integration produces an error.
Keywords	Enter optional keywords (tags). You can search on the connection keywords on the Connections page.
Description	Enter an optional description of the connection.
Share with other projects	**Note**: This field only appears if you are creating a connection in a project. Select to make this connection publicly available in other projects. Connection sharing eliminates the need to create and maintain separate connections in different projects. When you configure an adapter connection in a different project, the **Use a shared connection** field is displayed at the top of the Connections page. If the connection you are configuring matches the same type and role as the publicly available connection, you can select that connection to reference (inherit) its resources. See Add and Share a Connection Across a Project.

4. Click **Create**.

   Your connection is created. You're now ready to configure the connection properties, security policies, and (for some connections) access type.

5. Follow the steps to configure a connection.
   The connection property and connection security values are specific to each adapter. Your connection may also require configuration with an access type such as a private endpoint or an agent group.

6. Test the connection.

Configure Connection Properties

You can consume the SOAP-based catalog or REST web service catalog provided by OUAF in the Oracle Utilities Adapter.

1. Go to the **Connection Properties** section.

2. Enter the SOAP-based catalog or REST web service catalog URL exposed by an OUAF application in the **Catalog URL** field and click **OK**.

   The configured REST web service catalog should return only a list of REST inbound/outbound services:

   - Inbound services consist of REST Integrated Web Services (IWS).
   - Outbound services consist of the external system: the outbound message type for a real-time HTTP or JSON sender.

   The configured SOAP-based catalog should return only a list of SOAP inbound/outbound services:

   - Inbound services consist of SOAP Integrated Web Services (IWS).
   - Outbound services consist of the external system: the outbound message type for a real-time HTTP/SOAP message sender.

Configure Connection Security

Configure security for your Oracle Utilities Adapter connection by selecting the security policy.

1. Go to the **Security** section.
2. Select a security policy, and then complete the fields. You must already have created your client application to complete the necessary fields.

   The following security policy restrictions apply when configuring a Utilities Adapter connection with the trigger and invoke role on the Connections page:

   - If you select Basic Authentication, it can be used as a trigger and an invoke.
   - If you select the OAuth Resource Owner security policy, it can only be used as an invoke. Dragging the connection to the trigger area causes an exception error to be displayed. OAuth Resource Owner is only supported for REST on-cloud applications.
   - For existing integrations, the above restrictions do not apply when editing the Utilities Adapter in the Adapter Endpoint Configuration Wizard.
   - If you select OAuth Client Credentials as the security policy, it can be used as a trigger and an invoke. For triggering the integration with OAuth, the Oracle Utilities Adapter trigger connection must also use the OAuth Client Credentials security policy.
   - If you select Username Password Token as the security policy, it can be used as a trigger and an invoke.
   - If you use the OAuth Client Credentials using JWT Client Assertion security policy or OAuth using JWT User Assertion security policy, they can only be used with REST cloud-based connections.

Element	Description
Basic Authentication	Enter the following information: • **Username** • **Password**

Element	Description
OAuth Resource Owner Password Credentials **Note**: When using this security policy, do not configure the connectivity agent. This policy can only be used with REST-based connections.	This policy is for a catalog protected with OAuth 2.0 Token-Based Authentication. This enables you to consume an OpenAPI 3.*x* API protected with OAuth 2.0 Token-Based Authentication. This policy is useful when the Basic Authentication security policy is not sufficient. • **Access Token URI**: The OAuth server URL from which to obtain the access token. It is generally used by Oracle Identity Cloud Service server to identify where your application is registered (for example, `https://`*`idcs_hostname`*`/oauth2/v1/token`). • **Client ID**: The client identifier issued to the client during the registration process of the application with the Oracle Identity Cloud Service server. • **Client Secret**: The client secret issued to the client during the registration process of the application with the Oracle Identity Cloud Service server. • **Scope**: The scope for accessing the request. The scope enables you to specify which type of access you need. Scopes limit access for the OAuth token. They do not grant any additional permissions beyond that which you already possess (for example, `http:`*`hostname`*`:`*`port`*`/*`). • **Auth Request Media Type** : The format of the data you want to receive (for example, `application/x-www-form-urlencoded;charset=UTF-8`). This is an optional parameter that can be kept blank. • **Username**: The resource owner’s username (the application username). • **Password**: The resource owner's password (the application user password). • **Client Authentication**: It specifies how to send the client credentials in the request. The drop-down list provides two options: – **Send client credentials as basic auth header** – **Send client credentials in body** This is an optional parameter that if kept blank uses the default value of **Send client credentials as basic auth header**.
Username Password Token **Note**: Username Password Token security cannot be used with REST- based connections. An error is displayed when clicking **Test** to test the connection.	This policy is only supported for a SOAP-based catalog connection. • **Username** • **Password**

Element	Description
OAuth Client Credentials	• **Access Token URI** — The URL from which to obtain the access token. • **Client Id** — The client identifier issued to the client during the registration process. • **Client Secret** — The client secret. • **Scope** — The scope of the access request. Scopes enable you to specify which type of access you need. Scopes limit access for the OAuth token. They do not grant any additional permission beyond that which the user already possesses. • **Auth Request Media Type** — The format of the data you want to receive. This is an optional parameter that can be kept blank. • **Client Authentication** — You can optionally configure OAuth flows with client authentication. This is similar to the Postman user interface feature for configuring client authentication. – **Send client credentials as basic auth header**: Pass the client ID and client secret in the header as basic authentication. – **Send client credentials in body**: Pass the client ID and client secret in the body as form fields.
OAuth Client Credentials using JWT Client Assertion **Note**: This policy is typically used to invoke application-driven APIs where you use the client ID.	• **Access Token URI**: Enter the OAuth server URL from which to obtain the access token. It is generally used by the Oracle Identity Cloud Service server to identify where your application is registered. For example: `https://`*`idcs_hostname`*`/oauth2/v1/token)` • **JWT headers in JSON format**: Upload the JWT header file in JSON format. • **JWT payload in JSON format**: Upload the JWT payload file in JSON format. This payload contains `sub` as the client ID of the client application. • **JWT private key alias**: Enter the JWT private key alias. This is the same alias you specified when uploading the signing key certificate on the Certificates page. • **Scope**: (Optional) Enter the scope for accessing the request. The scope enables you to specify which type of access you need. Scopes limit access for the OAuth token. They do not grant any additional permissions beyond that which you already possess. • **Access token request**: (Optional) Enter the request to obtain the access token. The format you specify can vary by service provider.

Element	Description
OAuth using JWT User Assertion **Note**: This policy is typically used on behalf of a user.	• **Access Token URI**: The OAuth server URL from which to obtain the access token. It is generally used by the Oracle Identity Cloud Service server to identify where your application is registered. For example: `https://`*`idcs_hostname`*`/oauth2/v1/token` • **JWT headers in JSON format**: Upload the JWT header file in JSON format. • **JWT payload in JSON format**: Upload the JWT payload file in JSON format. This payload contains `sub` as the user name of the application. • **JWT private key alias**: Enter the JWT private key alias. This is the same alias you specified when uploading the signing key certificate on the Certificates page. • **Scope**: (Optional) Enter the scope for accessing the request. The scope enables you to specify which type of access you need. Scopes limit access for the OAuth token. They do not grant any additional permissions beyond that which you already possess. • **Access token request**: (Optional) Enter the request to obtain the access token. The format you specify can vary by service provider.

Configure the Endpoint Access Type

Configure access to your endpoint. Depending on the capabilities of the adapter you are configuring, options may appear to configure access to the public internet, to a private endpoint, or to an on-premises service hosted behind a fire wall.

- Select the Endpoint Access Type
- Ensure Private Endpoint Configuration is Successful

Select the Endpoint Access Type

Select the option for accessing your endpoint.

Option	This Option Appears If Your Adapter Supports ...
Public gateway	Connections to endpoints using the public internet.

Option	This Option Appears If Your Adapter Supports ...
Private endpoint	Connections to endpoints using a private virtual cloud network (VCN). **Note**: To connect to private endpoints, you must complete prerequisite tasks in the Oracle Cloud Infrastructure Console. Failure to do so results in errors when testing the connection. See Connect to Private Resources in *Provisioning and Administering Oracle Integration 3* and Troubleshoot Private Endpoints in *Using Integrations in Oracle Integration 3*.
Connectivity agent	Connections to on-premises endpoints through the connectivity agent. 1. Click **Associate agent group**. The Associate agent group panel appears. 2. Select the agent group, and click **Use**. To configure an agent group, you must download and install the on-premises connectivity agent. See Download and Run the Connectivity Agent Installer and About Creating Hybrid Integrations Using Oracle Integration in *Using Integrations in Oracle Integration 3*.

Ensure Private Endpoint Configuration is Successful

- To connect to private endpoints, you must complete prerequisite tasks in the Oracle Cloud Infrastructure Console. Failure to do so results in errors when testing the connection. See Connect to Private Resources in *Provisioning and Administering Oracle Integration 3*.
- When configuring an adapter on the Connections page to connect to endpoints using a private network, specify the fully-qualified domain name (FQDN) and *not* the IP address. If you enter an IP address, validation fails when you click **Test**.
- IPSec tunneling and FastConnect are not supported for use with private endpoints.

Test the Connection

Test your connection to ensure that it's configured successfully.

1. In the page title bar, click **Test**. What happens next depends on whether your adapter connection uses a Web Services Description Language (WSDL) file. Only some adapter connections use WSDLs.

If Your Connection...	Then...
Doesn't use a WSDL	The test starts automatically and validates the inputs you provided for the connection.

If Your Connection...	Then...
Uses a WSDL	A dialog prompts you to select the type of connection testing to perform: • **Validate and Test**: Performs a full validation of the WSDL, including processing of the imported schemas and WSDLs. Complete validation can take several minutes depending on the number of imported schemas and WSDLs. No requests are sent to the operations exposed in the WSDL. • **Test**: Connects to the WSDL URL and performs a syntax check on the WSDL. No requests are sent to the operations exposed in the WSDL.

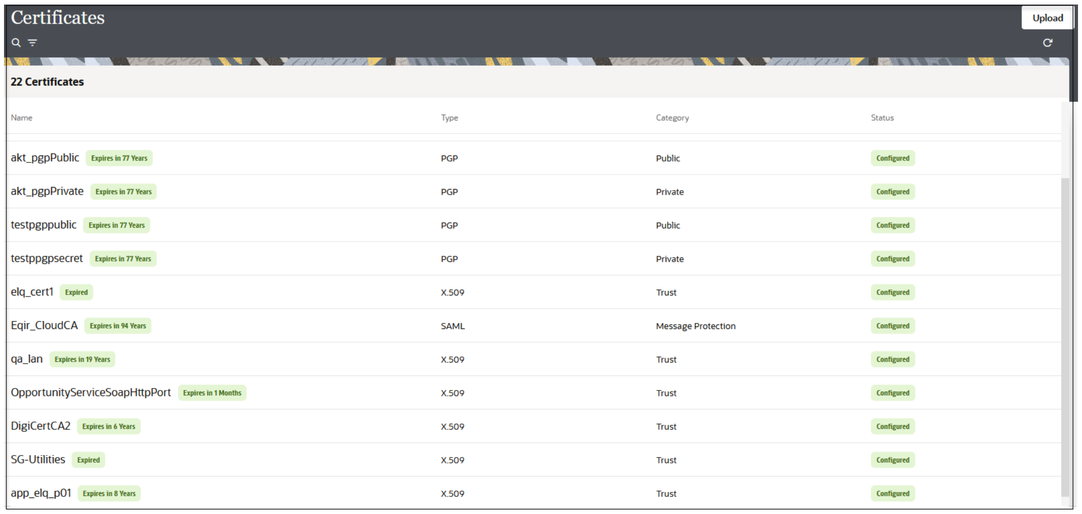
2. Wait for a message about the results of the connection test.
   - If the test was successful, then the connection is configured properly.
   - If the test failed, then edit the configuration details you entered. Check for typos and verify URLs and credentials. Continue to test until the connection is successful.
3. When complete, click **Save**.

Upload a Certificate to Connect with External Services

Certificates allow Oracle Integration to connect with external services. If the external service/ endpoint needs a specific certificate, request the certificate and then import it into Oracle Integration.

If you make an SSL connection in which the root certificate does not exist in Oracle Integration, an exception error is thrown. In that case, you must upload the appropriate certificate. A certificate enables Oracle Integration to connect with external services. If the external endpoint requires a specific certificate, request the certificate and then upload it into Oracle Integration.

1. Sign in to Oracle Integration.
2. In the navigation pane, click **Settings**, then **Certificates**.
   All certificates currently uploaded to the trust store are displayed on the Certificates page.
3. Click **Filter** to filter by name, certificate expiration date, status, type, category, and installation method (user-installed or system-installed). Certificates installed by the system cannot be deleted.



22 Certificates

Name	Type	Category	Status
akt_pgpPublic Expires in 77 Years	PGP	Public	Configured
akt_pgpPrivate Expires in 77 Years	PGP	Private	Configured
testpgppublic Expires in 77 Years	PGP	Public	Configured
testppgpsecret Expires in 77 Years	PGP	Private	Configured
elq_cert1 Expired	X.509	Trust	Configured
Eqir_CloudCA Expires in 94 Years	SAML	Message Protection	Configured
qa_lan Expires in 19 Years	X.509	Trust	Configured
OpportunityServiceSoapHttpPort Expires in 1 Months	X.509	Trust	Configured
DigiCertCA2 Expires in 6 Years	X.509	Trust	Configured
SG-Utilities Expired	X.509	Trust	Configured
app_elq_p01 Expires in 8 Years	X.509	Trust	Configured

4. Click **Upload** at the top of the page.

The Upload certificate panel is displayed.

5. Enter an alias name and optional description.
6. In the **Type** field, select the certificate type. Each certificate type enables Oracle Integration to connect with external services.
    - Digital Signature
    - X.509 (SSL transport)
    - SAML (Authentication & Authorization)
    - PGP (Encryption & Decryption)
    - Signing key

Digital Signature

The digital signature security type is typically used with adapters created with the Rapid Adapter Builder. See Learn About the Rapid Adapter Builder in Oracle Integration in *Using the Rapid Adapter Builder with Oracle Integration 3*.

1. Click **Browse** to select the digital certificate. The certificate must be an X509Certificate. This certificate provides inbound RSA signature validation. See RSA Signature Validation in *Using the Rapid Adapter Builder with Oracle Integration 3*.
2. Click **Upload**.

X.509 (SSL transport)

1. Select a certificate category.
    a. **Trust**: Use this option to upload a trust certificate.
        i. Click **Browse**, then select the trust file (for example, `.cer` or `.crt`) to upload.
    b. **Identity**: Use this option to upload a certificate for two-way SSL communication.
        i. Click **Browse**, then select the keystore file (`.jks`) to upload.
        ii. Enter the comma-separated list of passwords corresponding to key aliases.

        Note:

        When an identity certificate file (`.jks`) contains more than one private key, all the private keys must have the same password. If the private keys are protected with different passwords, the private keys cannot be extracted from the keystore.

        iii. Enter the password of the keystore being imported.
    c. Click **Upload**.

SAML (Authentication & Authorization)

1. Note that **Message Protection** is automatically selected as the only available certificate category and cannot be deselected. Use this option to upload a keystore certificate with SAML token support. Create, read, update, and delete (CRUD) operations are supported with this type of certificate.
2. Click **Browse**, then select the certificate file (`.cer` or `.crt`) to upload.
3. Click **Upload**.

PGP (Encryption & Decryption)

1. Select a certificate category. Pretty Good Privacy (PGP) provides cryptographic privacy and authentication for communication. PGP is used for signing, encrypting, and decrypting files. You can select the private key to use for encryption or decryption when configuring the stage file action.
    a. **Private**: Uses a private key of the target location to decrypt the file.
        i. Click **Browse**, then select the PGP file to upload.
        ii. Enter the PGP private key password.
    b. **Public**: Uses a public key of the target location to encrypt the file.
        i. Click **Browse**, then select the PGP file to upload.
        ii. In the **ASCII-Armor Encryption Format** field, select **Yes** or **No**.
            - **Yes** shows the format of the encrypted message in ASCII armor. ASCII armor is a binary-to-textual encoding converter. ASCII armor formats encrypted messaging in ASCII. This enables messages to be sent in a standard messaging format. This selection impacts the visibility of message content.
            - **No** causes the message to be sent in binary format.
        iii. From the **Cipher Algorithm** list, select the algorithm to use. Symmetric-key algorithms for cryptography use the same cryptographic keys for both encryption of plain text and decryption of cipher text. The following supported cipher algorithms are FIPS-compliant:
            - AES128
            - AES192
            - AES256
            - TDES
    c. Click **Upload**.

Signing key

A signing key is a secret key used to establish trust between applications. Signing keys are used to sign ID tokens, access tokens, SAML assertions, and more. Using a private signing key, the token is digitally signed and the server verifies the authenticity of the token by using a public signing key. You must upload a signing key to use the OAuth Client Credentials using JWT Client Assertion and OAuth using JWT User Assertion security policies in REST Adapter invoke connections. Only PKCS1- and PKCS8-formatted files are supported.

1. Select **Public** or **Private**.
2. Click **Browse** to upload a key file.
   If you selected **Private**, and the private key is encrypted, a field for entering the private signing key password is displayed after key upload is complete.
3. Enter the private signing key password. If the private signing key is not encrypted, you are not required to enter a password.
4. Click **Upload**.

4

Add the Oracle Utilities Adapter Connection to an Integration

When you drag the Oracle Utilities Adapter into the trigger or invoke area of an integration, the Adapter Endpoint Configuration Wizard is invoked. This wizard guides you through configuration of the Oracle Utilities Adapter endpoint properties.

The following section describe the wizard pages that guide you through configuration of the Oracle Utilities Adapter as a trigger or invoke in an integration.

Note:

The Oracle Utilities Adapter uses the logic of matching the description of the selected business object/service for edit/view mode to highlight the previously selected object/service. Therefore, the services description should remain the same in edge applications. Otherwise, by default the first service in the options from the catalog gets selected in the Adapter Endpoint Configuration Wizard.

Topics

- Add the Oracle Utilities Adapter as a Trigger Connection
- Add the Oracle Utilities Adapter as an Invoke Connection
- Integration Activation and Runtime

Add the Oracle Utilities Adapter as a Trigger Connection

When you drag the Oracle Utilities Adapter into the integration canvas as a trigger connection, the Adapter Endpoint Configuration Wizard is invoked. Based on your selections in the wizard, the following pages can be displayed.

Trigger Basic Info Page

Field	Description
What do you want to call your endpoint?	Provide a meaningful name so that others can understand the purpose of the connection. For example, `LinkedInTarget_update_status`. You can use English alphabetic characters, numbers, underscores, and dashes in the name. You cannot use: • Blank spaces (for example, `My FTP Connection`) • Special characters (for example, `#;83&` or `righ(t)now4`) • Multibyte characters
What does this endpoint do?	Enter an optional description of connection functionality.

Field	Description
What is the endpoint's relative resource URI? (Available only in the REST-based trigger (inbound) direction.)	Enter the endpoint's relative resource URI. The endpoint must begin with a /, followed by letters.

Note:

Starting with Release 23.08, the **Upload WSDL** option for trigger connections is no longer supported.

Trigger Request Page

This page enables you to select the external system to treat as the request object for this integration for OUAF applications.

Field	Description
Select a Business Object	Select the business object from the Oracle Utilities application to receive as a request that starts the integration.
Filter by object name	Enter the initial letters to filter the display of business objects.
Select Media Type (Available only for REST endpoints.)	JSON is displayed for selection.

For non-OUAF applications that are a REST-based endpoint, the trigger Request page is as follows:

Field	Description
Select a Request Object	Select a service request object to invoke.
Filter by object name	Enter the initial letters to filter the display of business services.
Request Media Type	Upon selecting the Operation, the supported media type will be displayed automatically. This can include the following values: • **JSON**: Denoting that the payload received is in JSON format. • **Multipart Mixed**: Indicates that an attachment is received with a JSON payload. For example, a CSV file accompanied by a JSON file that contains metadata information about the attachment.
Select the Operation	Select an operation from the published web service to perform on the service request object.

Trigger Response Page

This page enables you to select the external system to treat as the response object for this integration for OUAF applications.

Field	Description
Select a Business Object	Select the business object from the Oracle Utilities application to receive as a response from the integration.
Filter by object name	Enter the initial letters to filter the display of business objects.
Select Media Type (Available only for REST endpoints.)	JSON is displayed for selection.
Response Type	Select one of these options: • **Request-Response**: The default. The Oracle Utilities application waits until a response is received from the integration. This is also known as the request and response message exchange pattern. Disable the checkbox for asynchronous calls (also known as the one-way message exchange pattern). • **Send Faults**

For non-OUAF applications that are a REST-based endpoint, the trigger Response page is as follows:

Field	Description
Select a Response Object	Select the service response object from the Oracle Utilities application to receive as a response from the integration.
Filter by object name	Enter the initial letters to filter the display of business objects.
Response Media Type	Upon selecting the operation, the supported media type is displayed automatically. This can include the following values: • JSON: Indicates that the payload received is in JSON format. • Multipart Mixed: Indicates that an attachment is received, along with a JSON payload. For example, a CSV file accompanied by a JSON file that contains metadata information about the attachment.
Select the Operation	Select an operation from the published web service to perform on the service response object.
Response Type	Select one of these options: • **Request-Response**: The default. The Oracle Utilities application waits until a response is received from the integration. This is also known as the request and response message exchange pattern. Disable the checkbox for asynchronous calls (also known as the one-way message exchange pattern). • **Send Faults**

Summary Page

Field	Description
Summary	Displays a summary of the trigger (source) or invoke (target) configuration values that you defined on previous pages of the wizard. The information that's displayed can vary by adapter. For some adapters, the selected business objects and operation name are displayed. For adapters for which a generated XSD file is provided, click the **XSD** link to view a read-only version of the file. To return to a previous page to update any values, click the appropriate tab in the left panel or click **Back**.

Add the Oracle Utilities Adapter as an Invoke Connection

When you drag the Oracle Utilities Adapter into the integration canvas as an invoke connection, the Adapter Endpoint Configuration Wizard is invoked. Based on your selections in the wizard, the following pages can be displayed.

Invoke Basic Info Page

Field	Description
What do you want to call your endpoint?	Provide a meaningful name so that others can understand the purpose of the connection. For example, `LinkedInTarget_update_status`. You can use English alphabetic characters, numbers, underscores, and dashes in the name. You cannot use: • Blank spaces (for example, `My FTP Connection`) • Special characters (for example, `#;83&` or `righ(t)now4`) • Multibyte characters
What does this endpoint do?	Enter an optional description of connection functionality.

Invoke Operation Page

This page enables you to select the business service and operation to use for the target integration. Select the request or response payload type through which the endpoint can reply.

Field	Description
Select a Business Service	Select a business service to invoke.
Filter by service name	Enter the initial letters to filter the display of business services.
Select the Operation	Select an operation from the published web service.
Request Payload Type (Available only in the REST-based invoke (outbound) direction.)	Select the request payload format to use. • **JSON** • **Binary**: Enables the adapter to receive a file transfer using a content-type (for example, * \ * or application\octet-stream).

Field	Description
Response Payload Type (Available only in the REST-based invoke (outbound) direction.)	Select the response payload format to use. • **JSON** • **Binary**: Enables the adapter to send a file using a binary content-type (for example, * \ * or application\octet-stream)
Send Faults (Available only for REST endpoints.)	By default, **Send Faults** is enabled.

Summary Page

Field	Description
Summary	Displays a summary of the trigger (source) or invoke (target) configuration values that you defined on previous pages of the wizard. The information that's displayed can vary by adapter. For some adapters, the selected business objects and operation name are displayed. For adapters for which a generated XSD file is provided, click the **XSD** link to view a read-only version of the file. To return to a previous page to update any values, click the appropriate tab in the left panel or click **Back**.

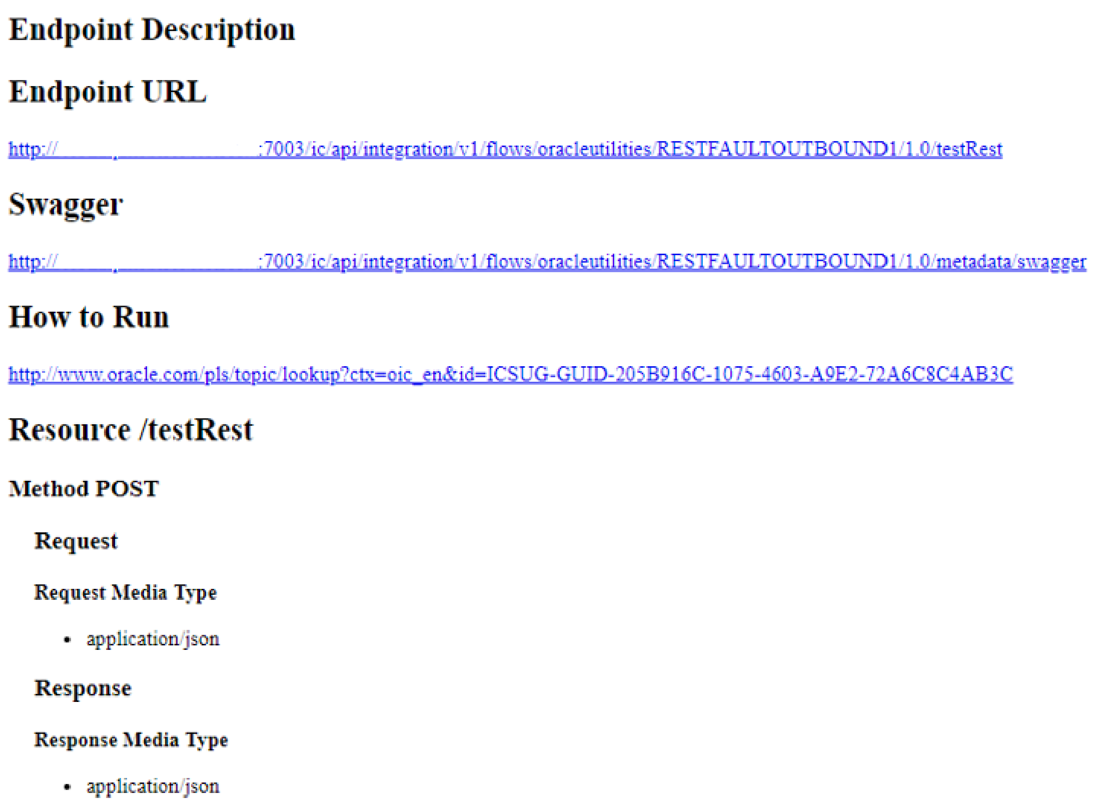
Integration Activation and Runtime

The activated integration provides an endpoint to trigger the integration.

A REST endpoint has the following information:

- REST endpoint URL
- Swagger definition URL
- Resource URI
- Request and response media type

For example:



Endpoint Description

Endpoint URL

http:// . :7003/ic/api/integration/v1/flows/oracleutilities/RESTFAULTOUTBOUND1/1.0/testRest

Swagger

http:// . :7003/ic/api/integration/v1/flows/oracleutilities/RESTFAULTOUTBOUND1/1.0/metadata/swagger

How to Run

http://www.oracle.com/pls/topic/lookup?ctx=oic_en&id=ICSUG-GUID-205B916C-1075-4603-A9E2-72A6C8C4AB3C

Resource /testRest

Method POST

Request

Request Media Type

- application/json

Response

Response Media Type

- application/json

5

Troubleshoot the Oracle Utilities Adapter

Review the following topics to learn about troubleshooting issues with Oracle Utilities Adapter.

Topics:

Additional integration troubleshooting information is provided. See Troubleshoot Oracle Integration in *Using Integrations in Oracle Integration 3* and the Oracle Integration Troubleshooting page on the Oracle Help Center.

Error Handling and Validations

Note the following issues when designing an integration with the Oracle Utilities Adapter.

- The Basic Authentication, OAuth Resource Owner Password Credentials, and OAuth Client Credentials security policies are currently supported by REST-based connections.
- The Username Password Token, Basic Authentication, and OAuth Client Credentials security policies are currently supported by SOAP-based connections.
- Dragging the Oracle Utilities Adapter connection to the trigger area of the integration canvas prompts you to enter the relative resource URI on the Basic Info page. The standard resource URI format starts with a `/`, followed by letters. If you enter any other URI format, an error message is displayed.
- Handling the cross-combination connection catalog error:
  If you create an integration using a SOAP-based connection, changing the connection to use a REST-based catalog has the following impact on that integration:
  - If the integration was already activated, there is no impact on the integration.
  - If the integration was not activated and you now attempt to activate the integration, it fails with the following error message:

    ```
    Activation Error:- This Integration was created using SOAP based
    connection
    but now connection changed to REST. Configure your connection to SOAP
    again or edit
    the integration for REST.
    ```

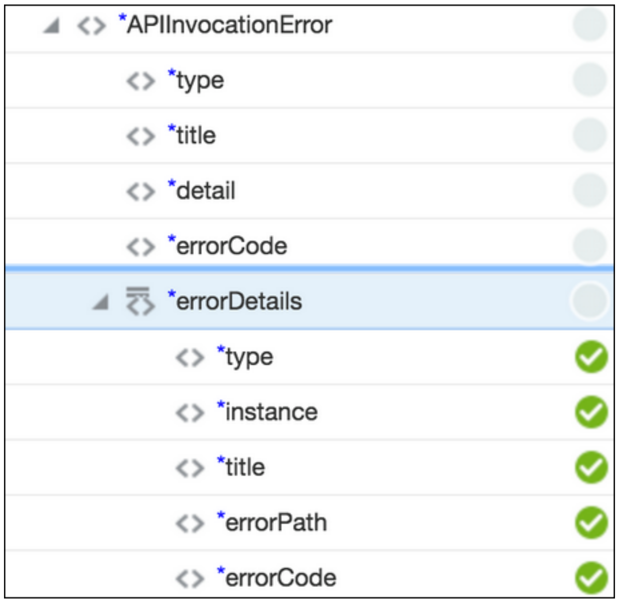
- If you create an integration using a REST-based connection, changing the connection to use a SOAP-based catalog has the same impact on that integration. The following error message is displayed:

  ```
  Activation Error:- This Integration was created using REST based
  connection but
  now connection changed to SOAP. Configure your connection to REST again or
  edit the
  integration for SOAP.
  ```

The Oracle Utilities Adapter uses the following strategy to handle errors in the invoke (outbound) and trigger (inbound) directions for REST endpoints.

- **Error Handling in the Invoke (Outbound) Direction**:
  The Oracle Utilities Adapter in the invoke (outbound) direction returns a standard `APIInvocationError` for any HTTP response that it receives with an error code. In addition, it also produces an `APIInvocationError` if a processing error occurs within the Oracle Utilities Adapter while preparing the request, calling the endpoint, or handling the response.

  The format of the `APIInvocationError` in the mapper is as follows.

  The `errorDetails` section contains the actual cause.

  You can handle the `APIInvocationError` with a fault handler in the orchestrated integration.

- **Error Handling in the Trigger (Inbound) Direction**:
  The Oracle Utilities Adapter in the trigger (inbound) direction exposes an HTTP endpoint that HTTP clients can request for using an HTTP request, and returns an HTTP response.

  If successful, the Oracle Utilities Adapter returns a success response. The Oracle Utilities Adapter returns an error response with an HTTP status belonging to the error family of codes depending on the situation.

  The Oracle Utilities Adapter also returns an error response with additional details about the error and possible steps for troubleshooting. The standard error response format is returned according to the configured response media type. The following is a sample JSON response structure:

```
{
  "type" : "http://www.w3.org/Protocols/rfc2616/rfc2616-
sec10.html#sec10.5.1",
  "title" : "Internal Server Error",
  "detail" : "An internal error occurred while processing the request.
Please see the fault details for the nested error details.",
  "o:errorCode" : "500",
  "o:errorDetails" : [ {
    "type" : "http://www.w3.org/Protocols/rfc2616/rfc2616-
sec10.html#sec10.4.1",
    "instance" : "{\n   \"error_message\" : \"Invalid request. Missing the
'origin' parameter.\",\n   \"routes\" : [],\n
    \"status\" : \"INVALID_REQUEST\"\n}\n",
    "title" : "Bad Request",
    "o:errorPath" : "GET http://maps.googleapis.com/maps/api/directions/
json?destination=Montreal returned a response status of
      400 Bad Request",
    "o:errorCode" : "APIInvocationError"
  } ]
}
```

The `o:errorDetails` section is reserved for the actual cause. The included prefix `o:` is based on Oracle standards.

Unmapped faults are propagated as system faults by Oracle Integration to the inbound Oracle Utilities Adapter. They may not communicate the appropriate details. Therefore, it is recommended that you define the fault pipelines.

java.net.ConnectException Error Message

If the error message `java.net.ConnectException: Connection refused: connect; No available router to destination.` appears, make sure the Oracle SOA server hosting the catalog is operating and accessible.

Unable to Connect to Edge Application at Run Time Error Message

If the following error message appears, make sure the connectivity and security credentials for the connection are correct. Validate the endpoint service from the application or Postman to troubleshoot the error.

```
Unable to connect to Edge Application at run time
```

See Create a Connection.

Unresponsive Agent Error Message

If the error message `No response received within response time out window of 120 seconds. Agent may not be running, or temporarily facing connectivity issues to`

`Oracle Messaging Cloud Service. Please check the health of the Agent in Agent.` appears and you are using the on premises agent, make sure the agent is operational and accessible.

Using the Swagger 2.0 REST Catalog with Oracle Utilities Adapter Version 24.04.0 or Higher

If you use the Swagger 2.0 REST catalog in a connection with Oracle Utilities Adapter version 24.04.0 or higher, the following error is displayed when you click **Test** for the connection.

```
Swagger 2.0 is no longer supported in Utilities Adapter. Please use openAPI
3.x REST catalog.
Please check Utilities Adapter documentation for more information.
```

If there is any existing integration using the Swagger 2.0 REST catalog, runtime won't be impacted. However, if you try to edit the design-time connection, retest the connection, refresh the metadata, refresh the artifacts, or reactivate, the integration fails. You must update the catalog to use the OpenAPI 3.*x* definition.

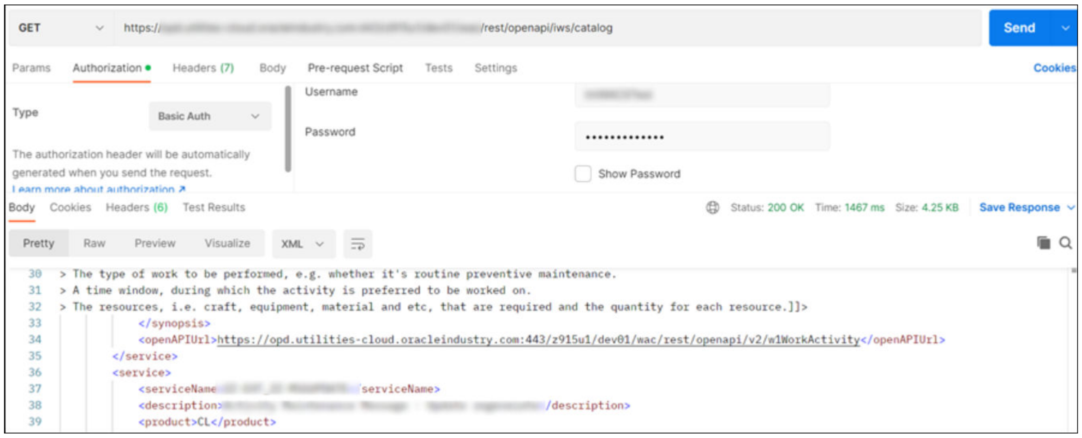
Note:

All Utilities Integration Accelerators on Oracle Integration 3 support OpenAPI 3.*x* catalog definitions. All non-OUAF (NMS) application catalogs are already on OpenAPI 3.*x*. For custom integrations, verify first if you are already using the OpenAPI 3.*x* catalog and your mappings are OpenAPI 3.*x*-compatible.

Differences Between the REST Swagger 2.0 Catalog and OpenAPI 3.*x* Catalog

To understand the difference between the two specifications:

1. Open Postman and `GET` the REST catalog response by providing valid credentials.



GET https://[redacted]/rest/openapi/iws/catalog

Type: Basic Auth

The authorization header will be automatically generated when you send the request.
Learn more about authorization ↗

Username: [redacted]
Password: ••••••••••••••
Show Password

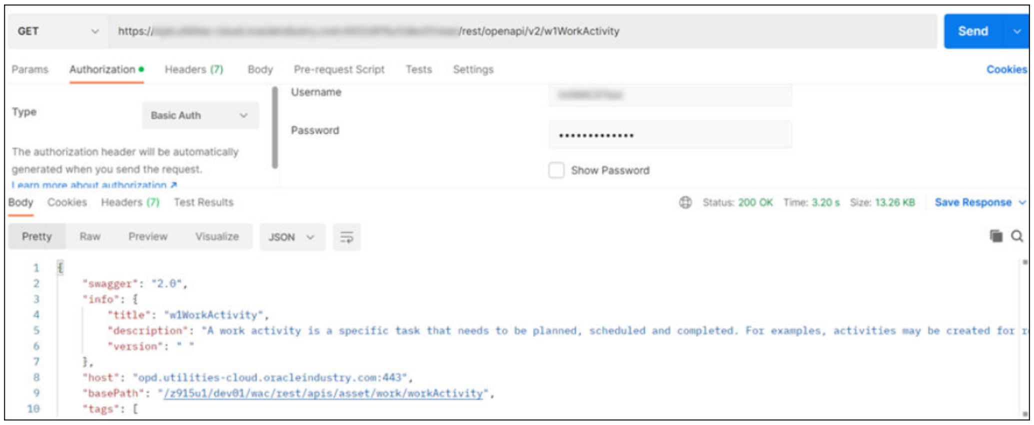
Status: 200 OK Time: 1467 ms Size: 4.25 KB

```
> The type of work to be performed, e.g. whether it's routine preventive maintenance.
> A time window, during which the activity is preferred to be worked on.
> The resources, i.e. craft, equipment, material and etc, that are required and the quantity for each resource.]]>
            </synopsis>
            <openAPIUrl>https://opd.utilities-cloud.oracleindustry.com:443/z915u1/dev01/wac/rest/openapi/v2/w1WorkActivity</openAPIUrl>
        </service>
        <service>
            <serviceName[redacted]serviceName>
            <description>[redacted]/description>
            <product>CL</product>
```

2. In the catalog response, check for any business service `openAPIUrl` field with the following format.

   ```
   <openAPIUrl>https://<hostname>:<port>/ouaf/rest/ouaf/openapi/v2/
   <servicename></openAPIUrl>
   ```

   - If `openAPIUrl` contains `/v2` in the URL, it's a Swagger 2.0 catalog. You can also access `openAPIUrl` from Postman or the Swagger editor using the `GET openAPIUrl` response. If the response contains `swagger : 2.0`, it's a Swagger 2.0 catalog.



GET https://[redacted]/rest/openapi/v2/w1WorkActivity

Params | Authorization | Headers (7) | Body | Pre-request Script | Tests | Settings | Cookies

Type: Basic Auth

The authorization header will be automatically generated when you send the request.
Learn more about authorization

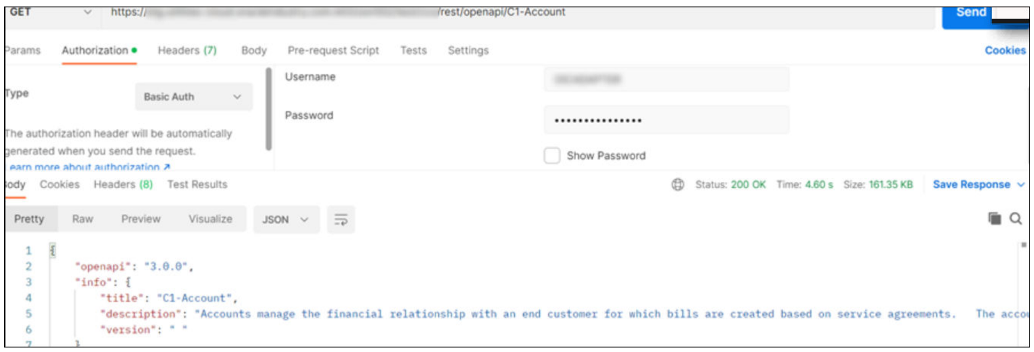
Username: [redacted]
Password: ••••••••••••••
Show Password

Body | Cookies | Headers (7) | Test Results — Status: 200 OK Time: 3.20 s Size: 13.26 KB — Save Response

Pretty | Raw | Preview | Visualize | JSON

```
{
    "swagger": "2.0",
    "info": {
        "title": "w1WorkActivity",
        "description": "A work activity is a specific task that needs to be planned, scheduled and completed. For examples, activities may be created for r
        "version": " "
    },
    "host": "opd.utilities-cloud.oracleindustry.com:443",
    "basePath": "/z915u1/dev01/wac/rest/apis/asset/work/workActivity",
    "tags": [
```

   - If the `openAPIUrl` field doesn't contain `/v2` in the URL, it's an OpenAPI 3.*x* catalog. You can also access `openAPIUrl` from Postman or the Swagger editor using the `GET openAPIUrl` response. If the response contains `openapi : 3.x.x`, it's an OpenAPI 3.*x* catalog.



GET https://[redacted]/rest/openapi/C1-Account

Params | Authorization | Headers (7) | Body | Pre-request Script | Tests | Settings | Cookies

Type: Basic Auth

The authorization header will be automatically generated when you send the request.
Learn more about authorization

Username: [redacted]
Password: ••••••••••••••••
Show Password

Body | Cookies | Headers (8) | Test Results — Status: 200 OK Time: 4.60 s Size: 161.35 KB — Save Response

Pretty | Raw | Preview | Visualize | JSON

```
{
    "openapi": "3.0.0",
    "info": {
        "title": "C1-Account",
        "description": "Accounts manage the financial relationship with an end customer for which bills are created based on service agreements.  The acco
        "version": " "
```

Replace REST Swagger 2.0 with the OpenAPI 3.*x* Catalog in an Integration

If your integration is using the REST Swagger 2.0 catalog in a connection, replace it with the OpenAPI 3.*x* catalog of the application. The catalog URL configured on the Connections page does not have any difference. Only the `openAPIUrl` of the specification for the inbound/outbound service in the catalog is different. The mappings in the design-time mapper are impacted after this change if the integration is edited.

If you are already using the OpenAPI 3.*x* catalog specification, verify it by opening the mapper for the corresponding Oracle Utilities Adapter. Verify if the root node begins with **components.schemas**.

- If it does, your catalog is an OpenAPI 3.*x*-supported catalog and the integration should work as is.
- If it's not, your mappings are old and using the Swagger 2.0 specification, which needs to be updated.

Otherwise, no change is needed and your custom integration should work without issues. If the mapper is using the Swagger 2.0 specification, it is recommended that you follow the knowledge article Steps To Fix Mapper Issue With Swagger2.0 To Open API 3.x to redo the mappings because the old mapping does not work in Oracle Integration 3.