Informatica

Informatica® Intelligent Cloud Services
November 2024

API Manager Guide

Informatica Intelligent Cloud Services API Manager Guide
November 2024

Publication Date: 2024-12-06

Table of Contents

Preface

Use *API Manager* to learn how to deploy, manage, and control the use of managed APIs that are built in Application Integration and custom APIs that are external to Informatica Intelligent Cloud Services.

CHAPTER 1

Introduction to API Manager

API Manager is a cloud-based service that an organization uses to deploy, manage, and control the use of APIs.

You can use API Manager to manage the APIs for enterprise services and processes that are built in Informatica Cloud Application Integration. You can also use API Manager to create and manage custom APIs that aren't built in Informatica Cloud Application Integration.

API Manager provides the following functionality:

- Seamless integration with Informatica Cloud Application Integration to manage APIs for Informatica Cloud Application Integration services using REST and SOAP protocols
- Creation and management of custom APIs that are external to the Informatica environment
- Rapid integration of APIs into organization applications through software development kit (SDK) packages
- API lifecycle management, including activating, deactivating, versioning, and deleting
- Access to API metadata and description for managed APIs
- Enforcement of authentication and authorization for managed APIs
- Enforcement of a rate limit policy, response caching policy, IP filtering policy, and privacy policy
- Grouping of APIs for easy management of APIs in the organization
- Management of organization access policies, including global rate limit and IP filtering policies
- API analytics, including dashboard, activity log, and event log

With API Manager, your organization ensures that internal or external users can safely and securely use the organization APIs. Administrators use API Manager to create managed APIs from Application Integration services, create custom APIs, and set policies and access authorization that control the usage of APIs.

By default, API consumers can access managed APIs through API Portal. You can choose to disable portal access for a managed API.

When you create a custom API, you can choose to make the API available on API Portal.

Administrators can monitor and analyze API usage with the API Analytics dashboard, and with activity and event logs. The Analytics dashboard provides a visual summary information about APIs, which includes trends in usage over time, APIs with the most invocations, and the most frequent users.

The activity log includes a comprehensive report of all API access attempts. The event log allows administrators to quickly identify and analyze unauthorized API access attempts, rate limit and IP filtering policy exceptions, and privacy policy leakages.

Workflow for creating and using an API

To create and use an API, perform the following steps:

1. Create an API. When API Manager creates an API, it activates it.
   a. To create a managed API for an Informatica Cloud Application Integration service, select the service.
   b. To create a custom API, provide the API URL.
2. Optionally, configure rate limit, response caching, IP filtering, and privacy policies for the API. If you create a managed API, you can also configure authentication and authorization.
3. Copy the API URL and provide the URL to API consumers. If you create a managed API and it uses token-based authentication, provide the required authentication details to API consumers.
   Alternatively, API consumers can use API Portal to access managed APIs and custom APIs. For more information about the API Portal, see the *API Portal* help.

API Manager user roles

A role is a collection of privileges that you can assign to users. To ensure that every user can access assets and perform tasks in an organization, assign at least one role to each user.

API Manager uses pre-defined roles to define access privileges for different types of assets and service features.

Before you use API Manager, ensure that you have an active Informatica Intelligent Cloud Services account, have assigned user roles through the Administrator, and have an API Manager license. To access API Manager, the user must be assigned both the **Admin** and **Service Consumer** roles, or both the **Deployer** and **Service Consumer** roles. If a new role is assigned, it is populated to the API gateway within 5 minutes.

You can assign the following types of roles to the API Manager users in Administrator:

API Manager Role	Role Type	Access Privileges
Admin	System-defined	Has full access to the service, including all asset privileges. Can access and perform all operations in the following pages in API Manager: - API Registry - API Groups - Policies - Analytics Can access and perform all operations in the following pages in API Portal: - API Registry - API Groups
APIM Administrator	Custom	Has full access to the service, including all asset privileges. Can access and perform all operations in the following pages in API Manager: - API Registry - API Groups - Policies - Analytics Can access and perform all operations in the following pages in API Portal: - API Registry - API Groups

Note: You can use only the system-defined Service Consumer role in API Manager. You can't use the cloned Service Consumer role.

For more information about registration and roles, see *User Administration*.

Note: A trial subscription includes access to API Manager with a trial license during the trial period.

URLs for managed APIs and custom APIs

When you create a managed API or custom API, API Manager creates the API URL based on the API properties. Each property is a segment in the managed API URL or custom API URL.

A managed API URL or custom API URL can contain the following segments:

- API domain name. API Manager uses the API domain name in the URL of the managed API or custom API to identify your organization. If you don't customize the API URL, API Manager uses the Informatica domain as the base URL followed by your organization's API domain name. If you customize the API URL, API Manager uses the API domain name as the base URL of all the managed APIs and custom APIs in your organization. The API domain name can contain up to 39 characters. For more information, see "Customized API URLs" on page 13.
- API name. When you create a managed API, you can enter a name or accept the default name. The API name can contain up to 50 characters.
- Version. When you create different versions of a managed API or custom API, API Manager adds the version to the URL. The version can contain up to 10 characters. For more information, see "Version management" on page 21.
- API Context. When you create a managed API or custom API, API Manager adds the API domain name, API name, and version to create the API context. The API context can contain up to 79 characters.

- Group context. When you create an API group, you can add a URL context to the group. API Manager adds the context to all the URLs of the APIs that belong to the group. The group context is independent from the API context. The API context can contain up to 66 characters. For more information, see Chapter 5, " API groups" on page 32.

Note: Query parameters in a managed API URL or custom API URL must not include spaces. Use `POST` to pass query parameters that include spaces.

Custom APIs

Custom APIs are APIs that aren't based on Informatica Cloud Application Integration processes. Custom APIs help you manage APIs that are external to the Informatica environment.

You can create and manage custom APIs that you add to your organization's list of APIs. You can make custom APIs available on API Portal.

You can't create custom APIs based on Informatica internal and Informatica public URLs.

To base a custom API on an encrypted URL, specify a URL that uses a certificate signed by a well-known Certificate Authority. You can't create a custom API for an HTTPS URL that uses a self-signed certificate or a certificate that isn't signed by a Certificate Authority.

Ensure that the API URL that you create the custom API for doesn't contain any potential vulnerability or malicious script.

API domain name

When you access API Manager for the first time, you are prompted to select an API domain name.

The API domain name identifies the organization and is used in the URLs created for the managed APIs and custom APIs of an organization. For example, the URL of the managed API of a service named `GetEmployee` might be `https://<base-url>/t/<API-domain-name>/GetEmployee`. You might want to use a subdomain of the organization domain.

Select the API domain name carefully. Any change of the API domain name after you start to use the system will result in the deletion of your organization API Manager settings, including managed APIs, custom APIs, policies, and analytics data.

Tip:

- You can customize the URLs of managed APIs and custom APIs so that the API domain name replaces the Informatica domain name as the base URL of the API. For more information, see "Customized API URLs" on page 13.
- On AWS, you can customize domain names in API Manager.

API versions

You can create multiple versions of a managed API or custom API.

For example, you can create multiple versions of a managed API that link to different Cloud Application Integration services. To run two Cloud Application Integration processes that are similar, instead of creating two managed APIs with different names, create two managed APIs that share the same name but have different versions.

If the managed API requires a JSON web token, generate a JSON web token for the new version of the managed API. If the managed API requires an OAuth 2.0 token, add the new version of the managed API to the OAuth 2.0 client and re-generate the token.

Accessing API Manager

Access API Manager through the Informatica Intelligent Cloud Services **My Services** page.

1. In the Informatica Intelligent Cloud Services login page, enter your user name and password.
2. Click **Log In**.
3. In the **My Services** page, select **API Manager**.

   API Manager appears.
4. If you are accessing API Manager for the first time, select an API domain name. On the **API Domain Name** window, define a unique domain name and click **Save**.

CHAPTER 2

API management

Use API Manager to manage the managed APIs and custom APIs in the organization. To manage a managed API, you select an Informatica Cloud Application Integration service and create a managed API for the service. To manage a custom API, you provide the API URL and create a custom API.

When API Manager creates the API, it activates it. You can copy the API URL and provide it to your API consumers. You can customize the URLs so that the API domain name replaces the Informatica domain name as the base URL of the API.

By default, all managed APIs can be accessed through API Portal. You can choose to disable portal access for a managed API. When you create a custom API, you can choose to make it available on API Portal. API consumers can use API Portal to access detailed information about APIs so that they can incorporate APIs into their applications. API consumers can also use API Portal to test and debug API execution. For more information about API Portal, see the *API Portal* help.

If you want to temporarily make a managed API or custom API unavailable, you can deactivate the API. Alternatively, you can hide an active managed API or custom API from API Portal without deactivating it. For example, during an API test phase, you can remove the display of the managed API from the portal until you are ready to release it to API consumers. When testing is complete, you can display the managed API in the portal again.

You can create different versions of a managed API or custom API to support multiple variations of an API. Create a new version of the API if you want to expose a variation of it without removing its existing variation. Versioning allows API consumers a grace period to adopt their applications to using the new version.

To stop running the service as a managed API or stop a custom API, you can delete the API. When you delete a managed API, the Informatica Cloud Application Integration service is not affected.

Use API Manager to generate SDK packages for rapid integration of managed APIs and custom APIs into organization applications. You can generate SDK packages that provide a set of resources to enable integration for Java, Android, Javascript, Nodejs, Python, Ruby-on-Rails, C#.NET, ASP.NET5, or C# applications.

Customized API URLs

You can customize the URLs of managed APIs and custom APIs so that the API domain name replaces the Informatica domain name as the base URL. The name can contain up to 39 characters.

For example: Your organization API domain is `sub.mydomain.com`, and the name of the managed API is Customers. When you create a managed API, API Manager creates one of the following API URLs, based on whether or not you configured API URL customization:

- If you don't customize the API URLs, API Manager creates the following URL:

  ```
  https://apigw-pod1.dm-us.informaticacloud.com/t/sub.mydomain.com/Customers
  ```
- If you customize the API URLs, API Manager creates the following URL:

  ```
  https://sub.mydomain.com/api/Customers
  ```

On AWS, you can customize domain names in API Manager.

Certificates for customized API URLs

Before you configure a customized API URL, you must provide Informatica Global Customer Support with a certificate for the organization API domain. You are limited to providing Informatica with two certificates.

Prerequisites for certificates

The certificate you use for API URL customization must meet the following conditions:

- The certificate is an SSL/TLS X.509 version 3 certificate. It contains a public key, the fully qualified domain name (FQDN) or IP address of your website, and information about your organization.
- The certificate is self-signed by your private key or by the private key of an issuing certificate authority (CA). If your certificate is signed by a CA, include the certificate chain when you import your certificate.
- The certificate, private key, and certificate chain is in PEM format.
- The private key is unencrypted. You cannot import a private key that is protected by a password or passphrase.
- The certificate is valid at the time of import. You can request at least one year of validity.
- The certificate specifies a cryptographic algorithm and a key size. Informatica supports the following algorithms:
  - 1024-bit RSA (RSA_1024)
  - 2048-bit RSA (RSA_2048)

Certificates for multiple sub-domains

If your organization uses multiple sub-domains, you can provide certificates in the following ways:

- Provide a certificate with a wildcard based domain name. For example, use the domain name `*.mydomain.com` to support all domain names that end with `.mydomain.com`, such as `dev.mydomain.com`, `qa.mydomain.com`, and `test.mydomain.com`.
- Provide a SAN (Subject Alternative Name) based certificate. SAN is an X.509 extension that allows multiple domains to be associated with a single certificate.

Before you start

To provide Informatica with a certificate for the organization API domain, perform the following tasks:

1. In your DNS, define routing from the organization API domain to the Canonical Name (CNAME) record of the Informatica domain. For example:

   ```
   https://apigw-pod1.dm-us.informaticacloud.com
   ```

   For more information about API domain names, see "API domain name" on page 10.
   For more information about DNS CNAME records, contact Informatica Global Customer Support.
2. Generate a certificate for the organization API domain.
3. Contact Informatica Global Customer Support and provide them with the certificate.

Informatica Global Customer Support will inform you when you can configure the customization.

Configuring API URL customization

After Informatica Global Customer Support informs you that you can configure API URL customization, perform the following steps:

1. In the **API Registry** page, click the Settings icon.
2. In the **API Domain Name** page, select **Use the API domain name as the default API URL**.
3. Click **Save**.

Certificate updates

Your organization creates certificates with a time limit for security. After expiration, the certificate is no longer valid and API calls are blocked automatically. Informatica cannot reissue the certificates. Therefore, your organization is responsible for monitoring certificate validity and for providing Informatica with an updated, signed certificate to replace the existing certificate before it expires.

Note: Your organization must create a new Informatica Global Customer Support ticket at least four weeks in advance of the existing certificate's expiration in order to update certificates in a timely manner and ensure their validity.

API Registry properties

The following table describes the **API Registry** page properties:

Property	**Description**
Icon	Identifies whether the entity is a service, managed API, or custom API: - : Designates a service. - : Designates a managed API. - : Designates a custom API.
Name	Name of the API. The name and version combination must be unique in the organization. The name can contain up to 50 characters, including any letter on the ASCII table, any digit, and the special characters . _ and -. The name can't contain spaces or any of the following characters: / ` " ' < > & : ~ ! @ # ; % ^ * + = { } | , \ $? ( ) . The name is part of the API URL.
Version	Version of the API. An API can have up to three versions. The version can contain up to 10 characters including any letter on the ASCII table, any digit, and the special characters . _ and -. The version can't contain spaces or any of the following characters: / ` " ' < > & : ~ ! @ # ; % ^ * + = { } | , \ $ ? ( ) Version can't be `t` because the segment `/t/` is reserved for Informatica internal use. Different versions of a managed API point to different Cloud Application Integration processes.
Protocol	The protocol of the service of the API: - REST - SOAP Informatica Cloud Application Integration services are published with both REST and SOAP endpoints.
Status	Status of the API: - Active ✓: The API is active. - Inactive ✗: The API is currently not available. - Service not available ✓: The service for the API is unavailable or deleted. If the service for which an API has been created is unavailable or deleted, the API is greyed out.
Authentication Method	Managed API authentication method. A managed API that invokes a Cloud Application Integration process that uses anonymous authentication also uses anonymous authentication. Managed APIs that use anonymous authentication don't require API consumers to authenticate. A managed API that invokes a Cloud Application Integration process that uses basic authentication can use one or more of the following authentication methods: - Basic. The API consumer provides an Informatica Intelligent Cloud Services user name and password for authentication. - OAuth 2.0. The API consumer provides an OAuth 2.0 access token to invoke the managed API. - JWT - JSON Web Token. The API consumer passes the JWT token as a bearer.
Group	The group that the API belongs to.
Description	Description of the service.

Creating and deleting a managed API

You can create a managed API for any Informatica Cloud Application Integration service. By default, the managed API is active and API Portal view for the API is enabled. The name that you assign to the API is part of the API URL.

1. On the **API Registry** page, select **Available Services**.

   The **API Registry** page list only services for which you can create a managed API.
2. Click to open the Actions menu of the service to create a managed API for and select **Create Managed API**.

   The **Create Managed API** page appears.
3. Enter a name for the API or accept the default name, and click **Create**.

   The name and version combination must be unique in the organization. The name can contain up to 50 characters, including any letter on the ASCII table, any digit, and the special characters . _ and -. The name can't contain spaces or any of the following characters: / ` " ' < > & : ~ ! @ # ; % ^ * + = { } | , \ $? ( ) .

   Note: A managed API name can't contain the keyword "_default" because "_default" is reserved for Informatica internal use.

   API Manager creates the managed API. The managed API is listed under **Managed APIs**. The service that you created the managed API from is no longer listed under **Available Services**.
4. By default, API Portal view is enabled. To disable API Portal view, clear **Available on API Portal**.

   The managed API does not show on the API Portal. API consumers can access the managed API with the Managed API URL.
5. Optionally, click **Add to Group** to add the API to a group.

   You can create a new group and add the API to the group or add the API to an existing group. For more information, see Chapter 5, " API groups" on page 32.
6. If you created the managed API from a service that uses basic authentication, select one or more of the following methods:

   - Basic. The API consumer provides an Informatica Intelligent Cloud Services user name and password for authentication.
   - OAuth 2.0. The API consumer provides an OAuth 2.0 access token to invoke the managed API.
   - JWT - JSON Web Token. The API consumer passes the JWT token as a bearer.

   For more information, see Chapter 6, "Authentication and authorization" on page 38.

   To delete a managed API, in the **API Registry** page, select the API, click to open the Actions menu, and select **Delete Managed API**. The Informatica Intelligent Cloud Services service on which the API is based is not affected.
   If the API is active, API Manager deactivates the API before it deletes it.

Creating and deleting a custom API

You can create a custom API that isn't based on an Informatica Cloud Application Integration service. The name that you assign to the custom API is part of the API URL.

When you create a custom API, the API URL that you specify must not be a URL that is restricted by Informatica and must not contain any potential vulnerability or malicious script. The URL must use a certificate that is signed by a well-known Certificate Authority.

1. On the **API Registry** page, click **Create Custom API**.

   The **Custom API** page appears.
2. Enter a name for the API or accept the default name.

   The name and version combination must be unique in the organization. The name can contain up to 50 characters, including any letter on the ASCII table, any digit, and the special characters . _ and -. The name can't contain spaces or any of the following characters: / ` " ' < > & : ~ ! @ # ; % ^ * + = { } | , \ $? ( ) .
3. Enter a version number for the API or accept the default version number.

   The name can contain up to 10 characters, including any letter on the ASCII table, any digit, and the special characters . _ and -. The name can't contain spaces or any of the following characters: / ` " ' < > & : ~ ! @ # ; % ^ * + = { } | , \ $? ( ) .
4. Enter a URL for the API.
5. Click **Test** to verify if the URL is valid and reachable.
6. Select a protocol from the list. You can select **REST** or **SOAP**.
7. Optionally, add a description.
8. To enable the API Portal view for the custom API and to allow API consumers to access it, select **Available on API Portal**. By default, the API Portal view is disabled.
9. Click **Save**.

   API Manager creates an API URL for the custom API. The custom API is listed on the **API Registry** page. Send the API URL to API consumers.
   You can configure API policies for custom APIs. You can't configure authorization and authentication for custom APIs.

   To delete a custom API, in the **API Registry** page, select the API, click to open the **Actions** menu, and select **Delete Custom API**.
   If the API is active, API Manager deactivates the API before deleting it.

After you create an API

After you create a managed API or custom API, you can perform the following actions:

- Copy the URL of the managed API. For managed APIs that invoke an Informatica Cloud Application Integration service, you can copy the Swagger or WSDL URL. For more information, see “Viewing managed API details and metadata” on page 18.
- For REST APIs, test the API. For more information, see “Interactively testing a managed API” on page 19.
- For managed APIs, generate and export an SDK package for the API. For more information, see “Generating and exporting an SDK package” on page 20.

- For managed APIs that invoke an Informatica Cloud Application Integration service that uses basic authentication, configure authentication authorization. For more information, see Chapter 6, “Authentication and authorization” on page 38.
- Rename the API. For more information, see “Renaming an API” on page 20.
- Deactivate the API. For more information, see “Deactivating and activating an API” on page 21.
- Configure a rate limit policy for the API. For more information, see “Configuring an API-specific rate limit policy” on page 28.
- Configure a response caching policy for the API. For more information, see “Configuring an API-specific response caching policy” on page 29.
- Configure an IP filtering policy for the API. For more information, see “Configuring an API-specific IP filtering policy” on page 30.
- Configure a privacy policy for the API. For more information, see “Configuring a privacy policy for an API” on page 31.
- Create a new version of the API. For more information, see “Version management” on page 21.

Viewing managed API details and metadata

After you create a managed API, you can view API details, including the API URL and the Swagger or WSDL URL, depending on the type of service from which you create the managed API.

Copy the URL of the managed API and provide it to API consumers to invoke the API. Copy the Swagger or WSDL URL and use it to view API metadata.

Tip:

- You can customize the URLs of managed APIs and custom APIs so that the API domain name replaces the Informatica domain name as the base URL of the API. For more information, see “Customized API URLs” on page 13.
- On AWS, you can customize domain names in API Manager.

1. In the **API Registry** page, click to select a managed API, or click to open the Actions menu and select **View API Details**.

   The **API details** window appears and displays relevant details.

   Note: API Manager **Service Name** is the Application Integration Process **Unique Name**, not the process name. The unique name differs from the process name when there are spaces or special characters in the process name, or when the same process name is given to processes in different folders in the Informatica Cloud organization.
2. To obtain the URL of the API, click **Copy URL**.

   Alternatively, you can obtain the API URL directly from the **API Registry** page. Perform the following actions:

   1. Select a managed API.
   2. Click to open the Actions menu, and select **Copy URL**.

   The URL is copied to the clipboard.
3. To obtain the URL to view metadata details for the managed API, in the **API details** window, select the available option:

- For a REST API, click **Copy Swagger URL**.
- For a SOAP API, click **Copy WSDL URL**.

The URL is copied to the clipboard.

4. Use one of the following methods to view metadata details for the managed API:
   - If the managed API uses anonymous or basic authentication, paste the URL in your browser.
   - If the managed API uses OAuth 2.0 or JSON Web Token (JWT) authentication, copy the URL to an application or service that supports token authentication such as Postman or SoapUI.

Interactively testing a managed API

You can interactively test a managed API created for a REST API in a Swagger interface. You can view the API URL, the HTTP status codes, the request parameters, and the response parameters. You can also execute the API for testing purposes, or get a sample cURL command.

1. In the **API Registry** page, click to select a managed API, or click to open the Actions menu and select **View API Details**.

   The API details window appears.
2. Select the **Swagger** tab.
3. If the managed API requires authentication, the **Authorization** dialog box appears. You might need to perform one of the following actions based on the authentication type that the managed API uses:
   - If the managed API uses JSON Web Token (JWT) authentication, no action is required. API Manager generates a JWT token and authenticates to the managed API.
   - If the managed API doesn't use JWT authentication and uses basic authentication, enter the username and password of a user who is authorized to access the managed API.
   - If the managed API uses only OAuth 2.0 authentication, enter the credentials of an OAuth 2.0 client that allows access to the managed API.
4. To expand the view in the **Swagger** tab, click the arrows in the upper right corner.
5. To view the API request body and response codes, click any button that displays an API method.

   For example, for an API with a POST method, a **POST** button is displayed. Click the **POST** button to view the API request body and response code. The input fields are displayed in the request body and the output fields are displayed in the response body with the data types and field descriptions.
6. To view the request body in JSON format, select **application/json**. To view the request body in XML format, select **application/xml**.
7. To test the API semantics, in the request body panel, perform the following steps:
   a. Click **Try it out**.
   b. Edit the request body. Replace any parameter type with a value.

      Note: If a query parameter in a managed API URL includes spaces, the access request fails. Use `POST` to pass a query parameter that includes spaces.
   c. To test the updated request body, click **Execute**.

      The **Server response** panel displays the response body, response headers, and request duration time.
   d. To clear the server response, click **Clear**.

e. To cancel the request body changes, click **Cancel**. To change the request body again, click **Edit**.

8. To view the request or response syntax, in the **Models** panel, click the right arrow near the request or response entry.

   The model request or response body is displayed. A red asterisk next to an element indicates a required element. The input fields are displayed in the request body and the output fields are displayed in the response body with the data types and field descriptions.

Generating and exporting an SDK package

You can generate and export an SDK package for a managed API. You can use the SDK package to integrate the managed API into your applications.

1. In the **API Registry** page, click to select a managed API.

   The API details window appears.
2. Select the type of client SDK package that you want to generate.
3. Click **Download**.

   The **API Registry** window appears.
4. Enter your API Manager user authorization details.

   The SDK package downloads to your host machine. To obtain information about the SDK package, read the `readme.md` file.

Renaming an API

When you rename a managed API or custom API, the API URL changes accordingly. Inform API consumers of the change in URL.

1. On the **API Registry** page, click the API.

   The API details window appears.
2. If the status of the API is Active, click **Deactivate**.
3. Click in the **API Name** field, name the API, and then click **Save**. The name and version combination must be unique in the organization. The name can contain up to 50 characters, including any letter on the ASCII table, any digit, and the special characters . _ and -. The name can't contain spaces or any of the following characters: / ` " ' < > & : ~ ! @ # ; % ^ * + = { } | , \ $? ( ) .

   API Manager saves the API with the new name. In the API URL field, the URL changes accordingly.
4. Click **Copy URL**. Send the new URL to API consumers.
5. To reactivate the API, click **Activate**.

Deactivating and activating an API

To temporarily disable a managed API or custom API, you can deactivate it. To use the API again, reactivate it.

For example, you might want to deactivate a managed API while you edit the Cloud Application Integration service that the API calls, or while you edit the settings of the managed API.

You can deactivate and activate an API on the **API Registry** page or in the **API details** window. Choose the method that is most convenient for you.

Deactivating and activating an API on the API Registry page

▶ On the **API Registry** page, click to open the Actions menu of the API. Select **Deactivate** or **Activate**, as applicable.

Deactivating and activating an API in the API details window

1. On the **API Registry** page, click the API.

   The **API details** window appears.

2. Click **Deactivate** or **Activate**, as applicable.

Version management

You can create multiple versions of a managed API or custom API, choose the default version, and delete a version.

When you create a version of an API, API Manager creates an additional API with the same name. Each version appears as a separate entry on the **API Registry** page. *Default* appears next to the entry of the default version on the **API Registry** page.

If a version of an API is not the default version, API Manager adds the version to the API URL of the API. API Manager doesn't add the version to the API URL of the default version.

API consumers can invoke all versions of an API with a URL that contains the version, including the default version. A URL that doesn't contain the version always invokes the default version.

In the API details window, the API URL of the default version doesn't include the version. If you choose a different version as the default version, API Manager adds the version to the API URL of the previously default version, and removes the version from the API URL of the current default version. Be sure to notify API consumers of the changes in API URLs.

When you create a new version of an API, you can choose whether or not API Manager applies the policies of the API version that the new version is based on to the new version. Policies include authentication method and API-specific policies. If you choose not to copy the policies from the original version, configure policies for the new version.

When you copy a policy of a managed API where token authentication is enabled, the tokens that the original version uses don't apply to the new version. Perform one of the following actions:

- If the original version uses OAuth 2.0 authentication, add the new version of the managed API to the OAuth 2.0 client and re-generate the token.

- If the original version uses JSON Web Token (JWT) authentication, create JWT tokens for the new version.

If an API has multiple versions, you can delete any version of the API other than the default version. To delete the default version of the API, choose a different version as the default version.

Creating an API version

Create a new version of a managed API or custom API. An API can have up to three versions.

1. In the **API Registry** page, select **Managed APIs**.

   The **API Registry** page lists all the available managed APIs and custom APIs.
2. Click to open the Actions menu of the API, and select **Create New Version**.

   The **Create New Version** page appears.
3. Enter a unique version.

   Version can contain up to 10 characters including any letter on the ASCII table, any digit, and the special characters . _ and -. It cannot contain spaces and any of the following characters: / ` " ' < > & : ~ ! @ # ; % ^ * + = { } | , \ $? ( )

   Note: Version cannot be "t" because it is reserved for Informatica internal use.
4. By default, the option **Copy policies from version <version>** is selected. You can deselect this option.

   If this option is selected, API Manager copies policies from the API version that the new version is based on, including the authentication method of the original version and the API-specific policies that are configured for the original version.
5. Select a service for the version from the list of available services.
6. Click **Create**.

   API manager creates a new version of the API.
   The version of the API appears in the list of APIs. The service that you created the version from no longer appears in the list of available services.
7. To set the version as the default version of the API, on the **API Registry** page, click to open the Actions menu of the new version and select **Set Version as Default**.

   Default appears next to the version on the **API Registry** page. API Manager adds the version to the API URL of the previously default version, and removes the version from the API URL of the current default version. Notify API consumers of the changes in API URLs.

Changing the default version of an API

When you change the default version of a managed API or custom API, API Manager adds the version to the API URL of the previously default version, and removes the version from the API URL of the current default version. Be sure to notify API consumers of the changes in API URLs.

You cannot change the version of an active API. To change the version of the API, deactivate it first. For more information, see "Deactivating and activating an API" on page 21.

1. In the **API Registry** page, select **Managed APIs**.

   The **API Registry** page lists all the available managed APIs and custom APIs.
2. Click to open the Actions menu of the API, and select **Set Version as Default**.
3. Click **OK** to confirm the action.

   The API is set as default. *Default* appears next to the version on the **API Registry** page.

CHAPTER 3

Organizational access policies

Organizational access policies are rules that the organization creates to enforce security and access rules on all managed APIs and custom APIs. The organization can enforce IP filtering access policies and determine the rate at which API requests can be made.

The IP filtering policy designates the range of computer IP addresses that are allowed to invoke or are denied permission to invoke APIs. The rate limiting policy controls the number of times any API can be invoked during a designated time period. Rate limit tiers determine the number of times that users can invoke the API during the designated time period.

The organizational rate limit policy controls the number of times API consumers in the organization can invoke an API during a designated time period. An organizational rate limit policy is assigned by default to all the APIs in the organization that are managed by API Manager. You can edit the default organizational rate limit policy.

In the **Access** tab of the **Policies** page, you can change the default rate limit policy settings, and add, edit, or delete an IP filtering policy. IP filtering policies are applied according to the order of the policies. The order of the policy determines its precedence.

You can also create rate limit policies and IP filtering policies for specific APIs and for API groups. API-specific rate limit and IP filtering policies override both the organizational and the group policies. API group rate policies override the organizational policies. For more information about API-specific policies, see Chapter 4, “API-specific policies” on page 26. For information about group policies, see “Group rate limit policy” on page 35.

When an API consumer attempts to access an API and access is denied due to an IP filtering policy, the HTTP response includes a `403 Forbidden` status code and the description `Invocation is prohibited due to organization policies`.

When an API consumer is denied access due to a rate limit policy, the HTTP response includes a `429 Too Many Requests` status code and the description `API rate limit reached`.

When an API consumer attempts to access an API and is denied due to a rate limit policy or an IP filtering policy, API Manager logs an event in the event log. For more information about the event log, see “Event log” on page 63.

Organizational rate limit policy

The organizational rate limit policy controls the number of times API consumers can invoke an API during a designated time period, for all the APIs in the organization that are managed by API Manager. You can't delete the rate limit rule, but you can change the default organizational rate limit for managed APIs and custom APIs.

The policy applies to all the APIs in the organization that don't have an API-specific rate limit policy or a group rate limit policy enabled. API-specific rate limit policies and API group policies override the organizational policy.

The organizational policy is defined by a rate limit tier. A tier is a logical entity that determines the number of times that users can invoke the API during a designated time period. You can update the organizational policy by editing the value of the default organizational rate limit policy tier or by selecting a different tier.

You assign the organizational rate limit policy tier to either be shared by all users or allocated per user. When you allocate the organizational tier to all users, the number of calls per time frame defined in the tier is divided between all users. When you allocate the tier per user, each user can make the defined number of calls within the defined time frame. You can also assign rate limit tiers to specific users in the organization.

The default rate limit for an API is 1,000 requests per minute. The maximal rate limit that you can define is 3,000 requests per minute.

Updating the organizational rate limit policy

Update the organizational rate limit policy on the **Access** tab of the **Policies** page.

1. To change the tier that defines the organizational rate limit policy, on the **Access** tab of the **Policies** page, in the **Rate Limit** panel, select the tier.
2. To edit the selected tier, perform the following steps:
   a. Click **Tier Setup**. The **Tier Setup** window opens.
   b. Select the tier, click to open the Actions menu, and select **Edit**.
      The **Edit Tier** window opens.
   c. Edit one or more of the following properties and then click **Save**:
      - Number of calls.
      - Number of seconds (Time Frame).
      - Description (optional).
   d. Click **Close**.
3. Select how to assign the organizational rate limit tier:
   - **Shared by all users**. All users share the rate limit that you select.
   - **Allocated per user**. Each user is allocated the rate limit that you select.
4. To assign user-specific tiers, expand the **User-specific Tiers** area and perform the following steps:
   a. Enter a user name in the **User Name** field.
   b. Select a rate limit tier from the **Tier** list and click **Add**.
   c. Repeat steps a and b to assign additional user-specific tiers as needed.
5. In the **Rate Limit** panel, click **Save**.

Organizational IP filtering policy

The organizational IP filtering policy consists of a set of access rules that determine IP addresses that are allowed or denied permission to invoke managed APIs and custom APIs. The policy applies to all the APIs in the organization. API-specific IP filtering policies and API group IP filtering policies override the organizational IP filtering policy.

When API Manager receives a request to invoke an API, it checks the IP address of the request against the rules in the policy, in the order that the rules are listed in. API Manager applies the first rule in the rules table that matches the request IP address to the request. You can change the order of the rules by moving them up or down.

To allow access to specific IP addresses and deny access to all other IP addresses, add the rules that allow access before the rule that denies access. To deny access to specific IP addresses and allow access to all other IP addresses, add the rules that deny access before the rule that allows access.

IP Filtering Policy Example

The following table lists rules that allow access to several IP address ranges, and deny access to all other IP addresses:

Access Type	From IP address	To IP address
Allow	9.10.11.12	9.10.25.27
Allow	5.6.7.8	5.6.15.17
Allow	1.2.3.4	1.2.12.14
Deny	0.0.0.0	255.255.255.255

Configuring an IP filtering policy

Configure an IP filtering policy that applies to all the managed APIs and custom APIs in the organization.

1. On the **Access** tab of the **Policies** page, on the **IP Filtering Policy** panel, select to allow or deny a range of addresses, and then fill in the IP range.
2. Add a description of the IP filtering rule. The description can contain up to 100 characters, including any special characters, and spaces.
3. Click **Add**.

   API Manager adds the rule to the **IP Filtering Rules** list.
4. Repeat steps 1 through 3 to create additional rules.
5. To move a rule up or down the list, rest on a row in the table and click the Action menu at the right end of the line. From the menu select **Move Up** or **Move Down**.
6. To change a rule from allowing an IP address range access to denying it access, or from denying access to allowing access, rest on a row in the table and click the Action menu at the right end of the line. From the menu select **Change to Deny All** or **Change to Allow All**.
7. To delete a rule, in the right-most column of the IP filtering policy row, rest on a row in the table and click the Action menu at the right end of the line. From the menu select **Delete**.
8. Click **Update**.

CHAPTER 4

API-specific policies

You can define policies to enforce security, access rules, and privacy for a specific managed API or custom API. You can also reduce redundant backend load by creating response caching policies for APIs.

For each API, you can configure the following policies:

- Rate limit policy. Controls the number of times API consumers can invoke the API during a designated time period, as defined by rate limit tiers.
- Response caching policy. Defines how long API Manager stores API responses in the cache.
- IP filtering policy. Designates the range of computer IP addresses that are allowed or that are denied permission to invoke the API.
- Privacy policy. Protects private information that is contained in API data.

API-specific rate limit policy

You can configure API-specific rate limit policies for managed APIs and custom APIs. The rate limit policy controls the number of times API consumers can invoke the API during a designated time period.

The API-specific rate limit policy overrides both the organizational and the group policies. For example, if the organizational rate limit is 10 invocations per second, and the API-specific rate limit is 20 invocations per second, API Manager rejects attempts to access the API after the 20 invocations per second limit is reached.

If an API-specific rate limit policy and a group rate limit policy aren't enabled for an API, API Manager applies the organizational policy to the API. The maximal rate limit that you can define is 3,000 requests per minute.

Rate limit tiers

When you configure a rate limit policy for an API, you define the policy by rate limit tiers.

A tier is a logical entity that determines the number of times that users can invoke the API during a designated time period. Different rate limit tiers prevent users from exploiting the system resources. You can create up to six tiers. Users can only use one tier for each invocation of an API.

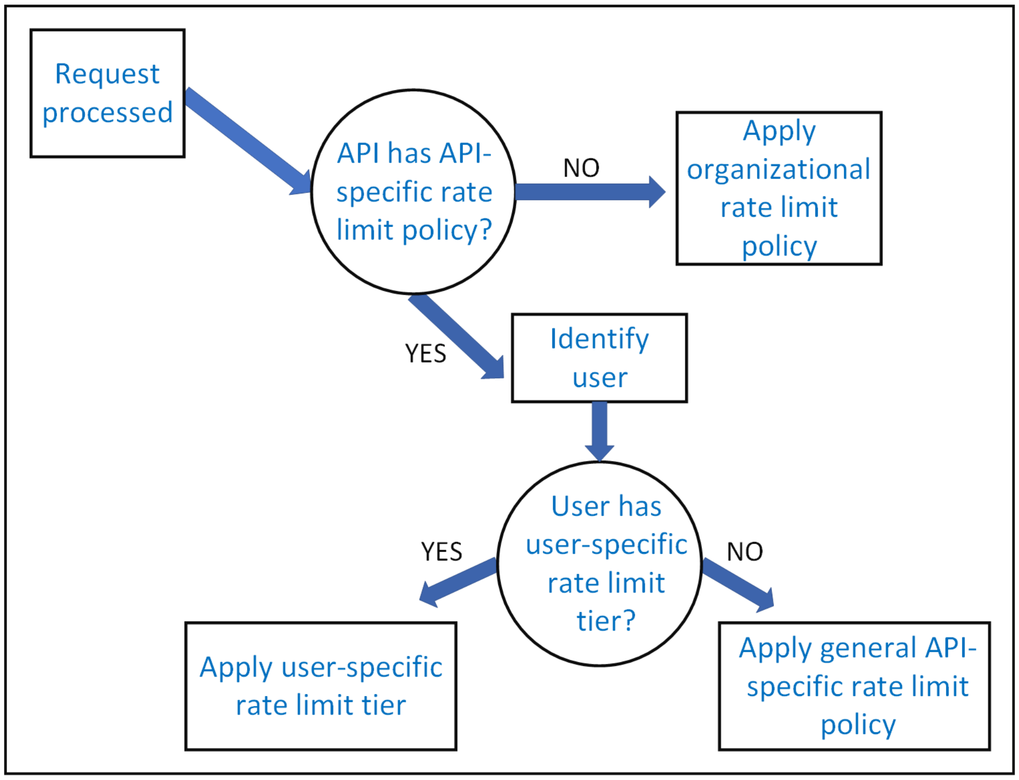
Assign the following rate limit tiers to APIs that use API-specific rate limit policies:

- A general API-specific rate limit tier that applies to all the consumers of the API. It affects all the users in your organization, except the users you assign a user-specific rate limit to. You can allocate the rate limit tier to all the users in your organization, or allocate it per user.
  When you allocate the general tier to all users, the number of calls per time frame defined in the tier is divided between all users. When you allocate the tier per user, each user can make the defined number of calls within the defined time frame.
- User-specific rate limit tiers. Each tier applies to the user that you assign the tier to.

You can also change the default organizational rate limit tier that you assign to all the managed APIs in the organization. For more information about organizational policies, see Chapter 3, "Organizational access policies" on page 23.

Processing requests

The following diagram shows the logical order for processing a request:


Request processed
API has API-specific rate limit policy?
NO
Apply organizational rate limit policy
YES
Identify user
User has user-specific rate limit tier?
YES
NO
Apply user-specific rate limit tier
Apply general API-specific rate limit policy

Access lock

API Manager locks access to users that exceed the number of allowed calls within the time frame that define the rate limit tier, based on the following logic:

- If a user exceeds the defined rate limit for the organizational rate limit policy or the API-specific rate limit tier, all the users from the same organization are locked from additional invocations of the same API using the same HTTP method.
- If a user exceeds the defined rate limit for a user-specific rate limit tier, the user is locked from additional invocations of the API using the same HTTP method.

API Manager logs an access exception in the event log. For more information about the event log, see "Event log" on page 63.

Creating tiers

Create tiers to assign to users of managed APIs and custom APIs.

1. On the **API Registry** page, select an API.

   The API details window appears.
2. Select the **Policies** tab.
3. Click **Tier Setup**.

   Enter the following properties to define the rate limit tier and click **Add**:

   - Name of tier. Assign tiers meaningful names, such as Gold, Silver, or Bronze. For example, configure Gold to be more permitting than Silver.
   - Number of calls.
   - Number of seconds (Time Frame).
   - Description (optional).

   You can create up to six tiers.
4. Click **Close**.

Configuring an API-specific rate limit policy

Configure a rate limit policy for a managed API or custom API by assigning a general API-specific tier that applies to all users that access the API. You can also assign tiers to specific users. User-specific tiers determine the access policies of the user that you assign them to and override the general API-specific tier.

1. On the **API Registry** page, select an API.

   The API details window appears.
2. Select the **Policies** tab.

   If no tiers are defined, click **Tier Setup** and create tiers.
3. Select **Enable API-specific rate limit policy**.
4. To assign a general API-specific rate limit policy, select a rate limit tier for the API from the **Tier** list. Select how to assign the tier:

   - To assign a rate limit tier that applies to all users, select **Shared by all users** and click **Save**. All users share the rate limit that you select.
   - To assign a rate limit tier that applies to each user individually, select **Allocated per user** and click **Save**. Each user is allocated the rate limit that you select.
5. To assign a user-specific rate limit policy to a specific API user, expand the **User-specific Tiers** area.
6. Enter the user name in the **User Name** field, select a tier from the **Tier** list and click **Add**.

   You can assign only one tier to each user.
7. Assign tiers to as many users as required.
8. Click **Save**.

   To disable the rate limit policy for the API, clear the option **Enable API specific rate limit policy** and click **Save**.

API-specific response caching policy

You can configure a response caching policy for a managed API or custom API. The response caching policy defines how long API Manager stores API responses in a cache repository.

When response caching is enabled for an API, API Manager saves responses to invocations of the API in the cache repository. Responses differ based on the input parameters that the API consumer sends to the API during invocation. When API consumers invoke the API, API Manager attempts to use saved responses in order to reduce time and resources. If a matching response is found in the cache, API Manager returns the response from the cache instead of calling the Cloud Application Integration process.

API Manager supports node-wise response caching. For example, assume that when an API consumer invokes an API that uses response caching, the call is processed by node A. If the subsequent invocation of the API is also processed by node A, API Manager returns the response from the cache. If the subsequent invocation is processed by a different node, say, node B, API Manager calls the Cloud Application Integration process again and saves the response in the cache of node B.

API Manager deletes responses from the cache after the defined timeout. When you disable the response caching policy for an API, API Manager deletes responses from the cache for the API.

Before you configure a response caching policy for an API, note the following considerations:

- The response cache can store a maximum of 200,000 bytes per API response, and up to 50 responses for a specific API.
- You can configure a maximum timeout of 3600 seconds.
- API Manager validates the IP filtering rules and basic authentication policy of the API before it stores the responses.
- API Manager stores API responses to GET requests that contain URL query parameters and to POST requests that contain an HTTP request body.
- API Manager stores API responses only to API invocations that return HTTP status code 200 to 299.
- API Manager matches current invocations of the API to previous invocations by comparing the query string parameters and the body of the request. It does not match the HTTP headers sent to the API.
- For responses that contain an empty JSON response payload, API Manager stores the `<jsonObject/>` string in the cache instead of the response payload.
- API Manager ignores `Cache-Control` directives in response headers. For example, if the response header contains `Cache-Control: no-store`, API Manager ignores it and stores the response in the cache.
- API Manager stores API responses that contain the media types `html`, `zip`, `excel`, `pdf`, `octerStream`, or `msWord` with media type `application/json`. If you download such a response through your browser, the browser doesn't append the appropriate extension to the response. For example, the browser doesn't append the extension `.zip` to responses that contain a `zip` media type.
- API Manager does not cache images. That is, it doesn't cache responses that contain media types `image/jpeg`, `image/png`, and `image/tiff`.

Configuring an API-specific response caching policy

Configure a response caching policy for a managed API or custom API on the **Policies** tab of the **API details** window to define how long API Manager stores API responses in the cache.

1. On the **API Registry** page, click to open the **Actions** menu of the API and select **View Details**.

   The API details window appears.
2. Select the **Policies** tab.

3. In the **Response Caching** section, select **Enable Response Caching**.
4. In **Cache Timeout**, enter a value between 1 and 3600, and then click **Save**.

   To disable the response caching policy for the API, clear the option **Enable Response Caching**, and then click **Save**.

API-specific IP filtering policy

You can configure an IP filtering policy for a managed API or custom API. The IP filtering policy designates the range of computer IP addresses that are allowed or that are denied permission to invoke the API.

If an IP filtering policy is not defined for an API, or if API Manager doesn't find any matches with the API-specific policy, API Manager applies the organizational IP filtering policy to the API.

When the policy is breached, API Manager logs an event in the even log. For more information about the event log, see “Event log” on page 63.

Configuring an API-specific IP filtering policy

Configure an IP filtering policy for a managed API or custom API on the **Policies** tab of the **API details** window to designate the range of computer IP addresses that are allowed or that are denied permission to invoke the API.

1. On the **API Registry** page, click to open the **Actions** menu of the API and select **Edit**.

   The **API details** window appears.
2. Click the **Policies** tab.
3. In the **IP Filtering Policy** section, select to allow or deny a range of addresses, and then fill in the IP range.
4. Add a description of the rule and click **Add**.

   The rule appears in the **IP Filtering Rules** list.
5. Add as many rules as required to define the policy, and then click **Save**. The order of the rules determines the precedence. The higher the rule is in the list, the higher the precedence.
6. To move a rule up or down the list and change the precedence, rest on a row and click the **Actions** menu at the right end of the line. From the menu, select **Move Up** or **Move Down**.
7. To delete a rule, in the right-most column of the **IP Filtering Rules** list, rest on a row in the table and click the **Actions** menu at the right end of the line. From the menu, select **Delete**.

   To disable the IP filtering policy for the API, delete all the policy rules.

API-specific privacy policy

You can configure a privacy policy for a managed API or custom API to protect private information that is contained in API data.

API Manager can issue warnings or block API requests and responses if the request or response payload contains the following information:

- Credit card number
- Email address
- IP address
- United States address
- United States phone number
- United States Social Security number

You can select different actions for requests and responses.

If you configure the policy to issue a warning or block requests and responses that contain certain types of information, API Manager logs an event in the event log when it receives requests and responses that contain information of that type.

For more information about the event log, see "Event log" on page 63.

Configuring a privacy policy for an API

Configure a privacy policy for a managed API or custom API on the **Privacy** tab of the **API details** window to protect private information that is contained in API data.

1. On the **API Registry** page, click to open the **Actions** menu of the API and select **View Details**.

   The API details window appears.
2. Select the **Privacy Policy** tab.
3. Select **Enable privacy policy**.
4. For each type of information that you want to protect, select the action to take for requests and for responses. You can select different actions for requests and responses.

   Select one of the following actions:

   - Warning. Issue a warning message in the event log that there was a privacy policy leakage in the request or the response. Don't block the request or response.
   - Block. Block the request or response and issue a warning message in the event log that the message was blocked because of a potential privacy policy breach in the request or the response.
5. Click **Save**.

CHAPTER 5

API groups

You can sort managed APIs and custom APIs into logical groups to more easily manage the APIs in the organization. Also, you can add a URL context to a group, create an IP filtering policy for a group, and generate a JSON web token for a group of APIs.

For example, create one group for APIs that handle salaries and a second group for APIs that handle HR updates, and generate a token for each group. This ensures that API consumers from the salary department cannot run APIs that handle HR updates, and API consumers from the HR department cannot run APIs that handle salaries.

An API can belong to only one group. When you add a URL context to an API group, API Manager adds the context to all API URLs of the APIs that belong to the group.

When you create an OAuth 2.0 client, you can choose the managed APIs in a group that the OAuth 2.0 client applies to. OAuth 2.0 tokens that you generate for the client can invoke all the managed APIs in the group for which OAuth 2.0 authentication is enabled. When you generate a JSON web token for an API group, you can use the generated token to invoke any managed API in the group for which JSON Web Token (JWT) authentication is enabled. For more information, see Chapter 6, "Authentication and authorization" on page 38.

API group context

If you define a context for the group, API Manager adds the context to the API URL of the APIs that belong to the group.

When you add a context to a group, consider the following guidelines:

- The group context must be unique.
- The context can contain up to 66 characters, including any letter on the ASCII table, non-Latin characters, any digit, and the special characters . _ and -. The context cannot contain spaces or any of the following characters: ` " ' < > & : ~ ! @ # ; % ^ * + = { } | , \ $? ( ) .

  Note: If you use customized API URLs, use only Latin characters in the group context.
- The context cannot be identical to the name of an API, or to the combination of an API name and any of its versions. Also, the combination of the group context, API name, and version of any API that belongs to the group cannot be identical to any other URL context in the organization. For example, if you want to create a group to contain version `b` of managed API `a`, the following restrictions apply:
  - The group context cannot be `/a/b`.
  - If `b` is the default version of `a`, the group context cannot be `/a` because the API context of the default version can either include the version or not, and thus can be either `/b/a` or `/a`.

- If `b` is not the default version of `a`, you can create a group with context `/a`, because the API context the non-default version always includes the version and is thus `/a/b`.

- The context is case sensitive.
- The context cannot contain a `/t/` segment because this segment is reserved for Informatica internal use. For example, the context cannot be `/a/t/b`.

API group management

Use the **API Groups** page to create, view, edit, and delete API groups. Use the **API Registry** page to add APIs to a group or remove APIs from the group.

You can add a managed API to a group when you create or edit the API. You can add a custom API to a group when you edit the API.

You can create a new group and add the API to the group or add the API to an existing group.

Creating an API group

Create an API group on the **API Groups** page and add APIs to the group. You can also add a context to the group and generate a JSON web token for managed APIs in the group.

1. In the **API Groups** page, click **New API Group**.

   The **New Group** page appears.
2. In the **Name** field enter a group name.

   The name is case sensitive and must be unique in the organization. The name can contain up to 255 characters, including any letter on the ASCII table, non-Latin characters, any digit, spaces, and the special characters . _ and -. The name cannot contain any of the following characters: / " ' < > & : ~ ! @ # ; % ^ * + = { } | , \
3. Optionally, enter a context for the group.

   Note: If an API name that contains non-Latin characters is added to a group with a URL context, access requests to the API fail. This condition applies only to organizations that use a custom domain name. To resolve this issue, use only Latin characters in the group context and API name.

   API Manager adds the context to the API URL of the APIs that belong to the group.
4. Optionally, enter a description of the group.
5. Click **Add APIs**.

   The **Add APIs to Group** page appears.
6. Select the APIs to add to the group and click **Add**.

   The **New Group** page lists the APIs that belong to the group in the **APIs in Group** section.
7. Click **Save**.

   API Manager creates the group. The **New Group** page appears, showing the group details.
8. Optionally, generate a JSON web token for the group.

   For more information, see "Generating JSON web tokens for managed APIs in a group" on page 37.

Adding APIs to a group

You can add APIs to an existing API group or create a group to add them to.

An API can belong to one group only.

1. On the **API Registry** page, select an API or multiple APIs, click to open the Actions menu, and select **Add to Group**.

   If API groups exist, the **Add to Group** dialog box appears. If there are no existing groups, the **New Group** dialog box appears. Go to Step 3.
2. If API groups exist, select a group from the **Group name** list and click **Add to Group**.

   API Manager adds the APIs to a group.
3. To add the APIs to a new group, click **New Group**, configure group details, and click **Save**. For more information, see "Creating an API group" on page 33.

   API Manager creates the group and adds the APIs to the group. If you defined a context for the group, API Manager adds the context to the API URL of all APIs that you add to the group.

Removing APIs from a group

You can remove APIs from an API group. You can only remove APIs from the same group at a time. If you select APIs from different groups and attempt to remove them, API Manager removes only one API from the registry.

1. On the **API Registry** page, select an API or multiple APIs from the same API group.
2. Click to open the Actions menu and select **Remove from Group**.
3. Click **Save**.

Viewing and editing API groups

Use the **API Groups** page to view and edit API groups.

To view or edit details of an API group, double-click the group name. The **Group Details** window appears and shows the APIs in the group in alphabetical order. The following table describes the properties that appear in the **Group Details** window.

Property	Description
Name	Name of the group. The name is case sensitive and must be unique in the organization. The name can contain up to 255 characters, including any letter on the ASCII table, non-Latin characters, any digit, spaces, and the special characters . _ and -. The name cannot contain any of the following characters: / " ' < > & : ~ ! @ # ; % ^ * + = { } | , \
Context	Group context. API Manager adds the context to the API URLs of the APIs in the group. Note the following guidelines: **Note:** If you add a context to an existing group, API Manager adds the context to the API URL of all the APIs in the group. Inform API consumers of changes to the API URLs.

Property	Description
Description	Description of the group. The description can contain up to 255 characters, including any letter on the ASCII table, non-Latin characters, any digit, spaces, and the special characters . _ and -.
JWT Access Token	JSON web token for the group. You can use the token to invoke any API in the group that uses JWT or JWT and Basic authentication.
APIs in Group	APIs that belong to the group. **Note:** If you remove an API from a group with a context, API Manager removes the group context from the API URL of the API that you remove from the group. Inform API consumers of changes to the API URLs.

Group rate limit policy

You can configure a group rate limit policy that applies to all the managed APIs and custom APIs in a group. The group rate limit policy controls the number of times API consumers can invoke the APIs in a group during a designated time period.

If an API in the group has an API-specific policy that is different than the group policy, API Manager applies the API-specific policy to the API.

If all the APIs in your organization have a group policy that is different than the organizational policy, API Manager applies the group policy to all the APIs in the group.

If you add managed APIs that use anonymous authentication to a group, the group rate limit policy applies to all users for each anonymous API in the group.

You can assign rate limit tiers to specific users. User-specific tiers determine the access policies of the user that you assign them to and override the general group tier. For more information, see "Rate limit tiers" on page 26.

Configuring a group rate limit policy

Configure a rate limit policy that applies to all the APIs in the group.

1. On the **API Groups** page, click to open the Actions menu of the API group and select **Edit**.

   The **Group Details** window appears.
2. Select the **Policies** tab.

   If no tiers are defined, click **Tier Setup** and create tiers. For more information, see "Creating tiers" on page 28.
3. In the **Rate Limit** area, select **Enable group rate limit policy**.
4. To assign a general group rate limit policy, select a rate limit tier from the **Tier** list. Select how to assign the tier:
   - To assign a rate limit tier that applies to all users, select **Shared by all users** and click **Save**. All users share the rate limit that you select.
   - To assign a rate limit tier that applies to each user individually, select **Allocated per user** and click **Save**. Each user is allocated the rate limit that you select.

5. Optionally, to assign a user-specific rate limit policy to a specific user, perform the following steps:
   a. Expand the **User-specific Tiers** area.
   b. Enter the user name in the **User Name** field, select a tier from the **Tier** list and click **Add**.

      You can assign only one tier to each user.
   c. Assign tiers to as many users as required.
6. Click **Save**.

   To disable the rate limit policy for the group, clear the option **Enable group rate limit policy** and click **Save**.

Group IP filtering policy

You can create an IP filtering policy that applies to all the managed APIs and custom APIs in a group. You can control user access to groups by allowing or denying access to specific IP addresses.

If an API in the group has an API-specific policy that is different than the group policy, API Manager applies the API-specific policy to the API.

If all the APIs in your organization have a group policy that is different than the organizational policy, API Manager applies the group IP filtering policy to all the APIs in the group.

If you add managed APIs that use anonymous authentication to a group, the group IP filtering policy applies to all users for each anonymous API in the group.

Configuring a group IP filtering policy

Configure an IP filtering policy that applies to all the APIs in a group.

1. On the **API Groups** page, click to open the Actions menu of the API group. Select **Edit**.

   The **Group Details** window appears.
2. In the **IP Filtering** section, select to allow or deny a range of addresses, and then fill in the IP range.
3. Add a description of the rule and click **Add**.

   The rule appears in the **IP Filtering Rules** list.
4. Add as many rules as required to define the policy, and then click **Save**. The order of the rules determines the precedence. The higher the rule is in the list, the higher the precedence.
5. To move a rule up or down the list and change the precedence, rest on a row and click the **Actions** menu at the right end of the line. From the menu, select **Move Up** or **Move Down**.
6. To delete a rule, in the right-most column of the IP filtering policy row, rest on a row in the table and click the **Actions** menu at the right end of the line. From the menu, select **Delete**.

   To disable the IP filtering policy for the group, delete all the policy rules.

Generating JSON web tokens for managed APIs in a group

You can generate JSON web tokens for managed APIs in a group on the **API Groups** page.

An API group can contain managed APIs with any authentication method. You can use the generated token to invoke any JWT or JWT and Basic authenticated managed API that is in the group.

For more information about configuring JWT tokens, see "JSON web token authentication" on page 46.

1. On the **API Groups** page, select an API group.

   The **Group Details** window appears.
2. Double-click the group name to open the group page.
3. In the **JWT Access Token** section, select an expiration date for the token and click **Generate**.

   API Manager generates a token for the managed APIs in the group.

Deleting an API group

When you delete a group, API Manager removes the group name and context from the API URL of the managed APIs and custom APIs that belong to the group. Inform API consumers of the changes in API URLs.

1. On the **API Groups** page, navigate to the group that you want to delete. and click **Actions**.
2. On the row that contains the group, click **Actions** and then click **Delete Group**.
3. To confirm that you want to delete the asset from the organization, click **Delete**.

   API Manager deletes the API group.

CHAPTER 6

Authentication and authorization

You can configure one or more authentication and authorization types for managed APIs that invoke a Cloud Application Integration process that uses basic authentication. You can't configure authentication for managed APIs that invoke a Cloud Application Integration process with an anonymous authentication or for custom APIs.

You can configure the following types of authentication and authorization:

- Basic. The user groups and users that are allowed access to the process in Informatica Cloud Application Integration can invoke the API. To invoke the managed API, API consumers authenticate with an Informatica Intelligent Cloud Services user name and password.
- OAuth 2.0. You create an OAuth 2.0 client in API Manager. API consumers use the client credentials to generate an OAuth 2.0 authorization token that they use to invoke the API. API Manager supports the client credentials grant type for OAuth 2.0 authentication.
- JSON Web Token (JWT). You generate a token using API Manager or API Portal. API consumers use the generated token to invoke the API.

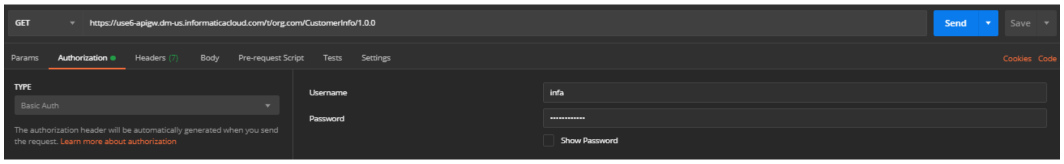
When you create a managed API that invokes a Cloud Application Integration process that uses basic authentication, the authentication method of the managed API is basic. You can then add and remove authentication methods from the managed API. When you change the authentication method, the managed API might become unavailable to users for a very short period of time.

Basic authentication

You can enable basic authentication for managed APIs that invoke a Cloud Application Integration process that uses basic authentication. User groups and users that are allowed access to the process in Informatica Cloud Application Integration can invoke the API.

To invoke the managed API, API consumers authenticate to the API with an Informatica Intelligent Cloud Services user name and password.

The following image shows an API invoked through Postman with the authorization type set to **Basic Auth**, and the user name and password specified:



GET https://use6-apigw.dm-us.informaticacloud.com/t/org.com/CustomerInfo/1.0.0

TYPE: Basic Auth

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

Username: infa

Password: ••••••••••••

OAuth 2.0 authentication and authorization

You can enable OAuth 2.0 authentication for managed APIs that invoke a Cloud Application Integration process that uses basic authentication.

OAuth 2.0 is a protocol for authorization that provides specific authorization flows for web applications and helps in the secure transmission of information between API consumers and web services such as Informatica Cloud Application Integration service APIs.

API Manager supports the client credentials grant type for OAuth 2.0 authentication.

Enabling OAuth 2.0 authentication

To enable OAuth 2.0 authentication, you perform the following tasks:

1. Create an OAuth 2.0 client. Specify credentials of an organization user with access to run managed APIs, select managed APIs or managed API groups that can use the client for authentication, and generate client credentials.
2. Send the following details to consumers of the managed APIs that the client applies to:
   - Informatica Intelligent Cloud Services OAuth 2.0 server URL.
   - Client credentials.

Invoking a managed API where OAuth 2.0 authentication is enabled

To invoke a managed API where OAuth 2.0 authentication is enabled, API consumers perform the following tasks:

1. Authenticate against the Informatica Intelligent Cloud Services OAuth 2.0 server and use the OAuth 2.0 client credentials to generate an OAuth 2.0 authorization token.
2. Use the OAuth 2.0 authorization token to invoke the API.

Creating an OAuth 2.0 client

Create an OAuth 2.0 client that enables managed API consumers to invoke managed APIs where OAuth 2.0 authentication is enabled.

1. On the **Policies** page, select the **Authorization** tab.
2. Click **Add OAuth 2.0 Client**.

   The **Add OAuth 20.0 Client** wizard appears.
3. Enter the user name and password of an organization user with the Service Consumer role and click **Next**.

   The **Details** step appears.
4. Enter a name for the client. The name is case sensitive and must be unique in the organization.

   The name can contain up to 32 characters, including any letter on the ASCII table, non-Latin characters, any digit, spaces, and the special characters . _ and -. The name cannot contain any of the following characters: / " ' < > & : ~ ! @ # ; % ^ * + = { } | , \
5. Optionally, enter a description of the client.

   The description can contain up to 1,024 characters, including any letter on the ASCII table, non-Latin characters, any digit, spaces, and the special characters . _ and -.
6. Enter a timeout value in minutes for the access token.

The minimum value is 5 minutes and the maximum value is 1440 minutes or 24 hours. Default is 60 minutes.

After a token times out, you cannot use it. You must regenerate the token.

7. Click **Next**.

   The **Resources** step appears.

8. Select managed APIs that the OAuth 2.0 client applies to using one of the following options:

   - All API Resources. The OAuth 2.0 client applies to all the managed APIs in the organization.
   - APIs and API Groups. Select managed APIs and managed API groups that the OAuth 2.0 client applies to. You can select up to 50 managed APIs and up to 10 groups.

   Click **Next**.

   API Manager creates the client. The **Generated Credentials** step appears.

9. Copy the client credentials and send them to API consumers using one of the following methods:

   - Click **Copy** next to **OAuth 2.0 Client ID** and **OAuth 2.0 Client Secret** to copy the credentials as plain text. API consumers use the client credentials in applications and software packages where you enter each detail separately.

     Note: You cannot copy the client secret after you exit the wizard.

   - Click **Copy Basic Authorization Header Value** to copy the credentials as an authorization header value. API consumers use the value in applications and software packages where you enter the client credentials as a value in a Basic authorization header. For example, if the value you copy is 4879857439857349857, API consumers enter the following authorization header: `Basic 4879857439857349857`.

     Note: You can't use the `DOCTYPE` header in XML attachments.

10. Click **Finish**.

11. On the **Authorization** tab, click **Copy URL** and send the OAuth 2.0 server URL to API consumers.

Managing OAuth 2.0 clients

After you create an OAuth 2.0 client, you can view, edit, and delete it.

1. On the **Policies** page, click the **Authorization** tab.

   The **OAuth 2.0 Clients** section displays all the OAuth clients and their details.

2. Perform one of the following tasks:

   a. To view the details of an OAuth 2.0 client, double-click the client.

   b. To edit an OAuth 2.0 client, click the **Actions** menu in the row that contains the OAuth 2.0 client, and select **Edit**. Make the required changes and click **Save**.

   c. To delete an OAuth 2.0 client, click the **Actions** menu in the row that contains the OAuth 2.0 client, and select **Delete Client**.

      A message appears prompting you to confirm the deletion. Click **Delete** to proceed with the deletion, or click **Cancel** to cancel the deletion.

Regenerating an OAuth 2.0 client secret

You can regenerate an OAuth 2.0 client secret if needed. When you regenerate the client secret, API Manager disables the current client secret.

1. On the **Policies** page, click the **Authorization** tab.

   The **OAuth 2.0 Clients** section displays all the OAuth clients and their details.

2. To regenerate the secret, click the **Actions** menu in the row that contains the OAuth 2.0 client, and select **Regenerate Secret**.

   Alternatively, you can double-click an OAuth 2.0 client, and click **Regenerate Secret**.

3. Enter the password and click **Next**.

   API Manager regenerates the client secret and authorization header value. You can copy the new values and send them to the API consumers.

Enabling and disabling OAuth 2.0 clients

When you create an OAuth 2.0 client, it is enabled by default. You can disable an OAuth 2.0 client if needed. API consumers cannot use disabled OAuth 2.0 clients for authentication.

1. On the **Policies** page, click the **Authorization** tab.

   The **OAuth 2.0 Clients** section displays all the OAuth clients and their details.

2. To enable or disable a client, click the **Actions** menu in the row that contains the OAuth 2.0 client, and select **Enable Client** or **Disable Client**.

   Alternatively, you can double-click an OAuth 2.0 client, and click **Enable Client** or **Disable Client**.

Note: When there are multiple bad attempts to get the token or when there are other violations for an OAuth 2.0 client, the OAuth2 Identify Provider sets the status of the client to locked. API consumers cannot use locked OAuth 2.0 clients for authentication.

How API consumers invoke an API with OAuth 2.0 authentication

To invoke a managed API where OAuth 2.0 authentication is enabled, API consumers generate an OAuth 2.0 authorization token and send the token to the managed API.

The following sections describe the stages of invoking a managed API that uses OAuth 2.0 authentication:

Generating an OAuth 2.0 authorization token

To generate the token, API consumers authenticate to the IDMC OAuth 2.0 server using the server URL and the OAuth 2.0 client credentials that you send to the API Portal administrator.

API consumers use one of the following methods to provide the client credentials to the OAuth 2.0 server, based on the application or software package that they use to invoke the API:

- Enter the authentication header value as **Basic**. Select **Client Credentials** in the **Grant Type** field and enter the URL in the **Access Token URL** field.
- Enter the OAuth 2.0 client ID and secret separately, as plain text.

For example, in Postman, enter the details as follows and add them to the request body as URL-encoded data:

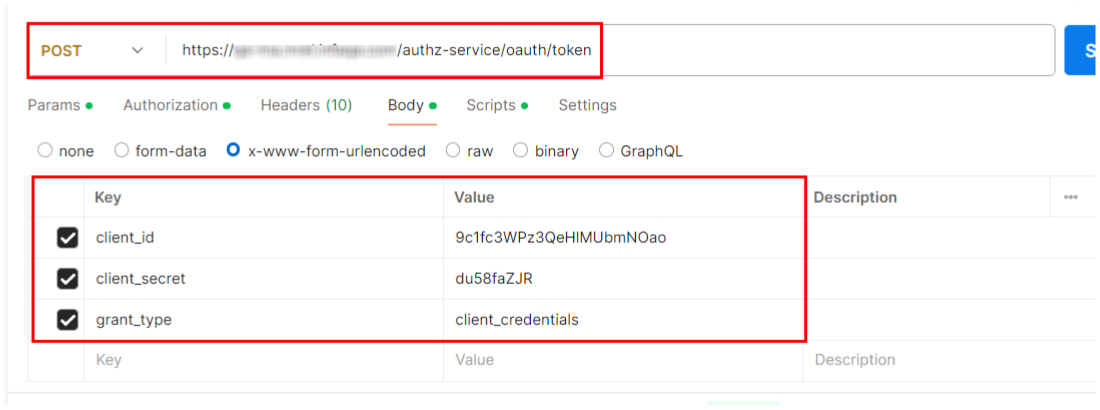
Key	Value
client_id	Client name
client_secret	Client secret
grant_type	client_credentials

You can find the access token URLs on the **Authorization** tab in the **Configuration** page.

An access token POST URL has the following format:

`{protocol}://{Host_URL}/authz-service/oauth/token`

The following image shows a sample POST URL and other details:



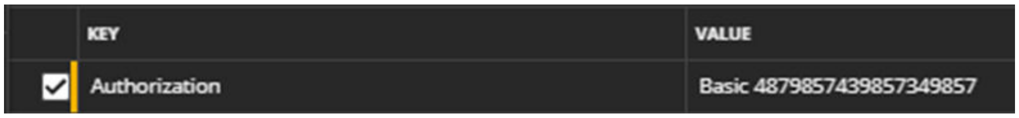
POST https://[redacted]/authz-service/oauth/token

none | form-data | [✓ x-www-form-urlencoded] | raw | binary | GraphQL

	Key	Value	Description
✓	client_id	9c1fc3WPz3QeHIMUbmNOao	
✓	client_secret	du58faZJR	
✓	grant_type	client_credentials	
	Key	Value	Description

- Enter the **client_id** and the **client_secret** as authentication header values encoded in a combined Base64 Basic authorization header. Select **Client Credentials** in the **grant_type** field add them to the request body as URL-encoded data, and enter the URL in the **Access Token URL** field.

The following image shows an API invocation through Postman with a Basic authorization header:

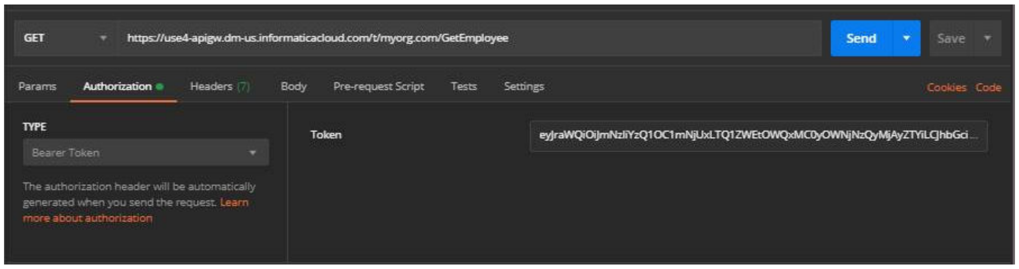


	KEY	VALUE
✓	Authorization	Basic 4879857439857349857

Sending the token to the managed API

API consumers pass the token that they receive from the OAuth 2.0 server to the managed API as an Authorization header with the prefix `Bearer` followed by the token.

The following image shows an API invoked through Postman with a Bearer Token authorization type and the token that the API consumer entered:



https://use4-apigw.dm-us.informaticacloud.com/t/myorg.com/GetEmployee

TYPE: Bearer Token

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

Token: eyJraWQiOiJmNzliYzQ1OC1mNjUxLTQ1ZWEtOWQxMC0yOWNjNzQyMjAyZTYiLCJhbGci...

Python 3 example: Invoke a managed API with OAuth 2.0 authentication

You can invoke a managed API where OAuth 2.0 authentication is enabled in Python 3.

In order to invoke a managed API with the OAuth 2.0 authentication method, API consumers must request an OAuth 2.0 token from the Informatica Intelligent Cloud Services OAuth 2.0 server.

You can use any OAuth 2.0 library, tool, or programming language to run the OAuth 2.0 authentication sequence. Before you run the OAuth 2.0 authentication, verify that you have the following information:

- URL of the Informatica Intelligent Cloud Services OAuth 2.0 server.
- OAuth 2.0 client ID and secret with permissions to run the managed API.

The following example shows the codes used for invoking a managed API with OAuth 2.0 authentication in Python 3:

```
import sys
import requests
import json
import logging
import time

logging.captureWarnings(True)

test_api_url = "https://apigw-pod1.dm-us.informaticacloud.com/t/apim.usw1.com/
get_employee_details"

##
##    function to obtain a new OAuth 2.0 token from the authentication server
##
def get_new_token():

auth_server_url = "https://dm-us.informaticacloud.com/authz-service/oauth/token"
client_id = 'Jl88QzqE3GYvaibOVb1Fx'
client_secret = '9xy23jdl'

token_req_payload = {'grant_type': 'client_credentials'}

token_response = requests.post(auth_server_url,
data=token_req_payload, verify=False, allow_redirects=False,
auth=(client_id, client_secret))

if token_response.status_code !=200:
            print("Failed to obtain token from the OAuth 2.0 server", file=sys.stderr)
            sys.exit(1)

            print("Successfuly obtained a new token")
            tokens = json.loads(token_response.text)
            return tokens['access_token']

##
##     obtain a token before calling the API for the first time
##
token = get_new_token()
```

```
while True:

##
##   call the API with the token
##
api_call_headers = {'Authorization': 'Bearer ' + token}
api_call_response = requests.get(test_api_url, headers=api_call_headers, verify+False)

##
##
if    api_call_response.status_code == 401:
            token = get_new_token()
else:
print(api_call_response.text)

time.sleep(30)
```

Java example: Invoke a managed API with OAuth 2.0 authentication

You can invoke a managed API where OAuth 2.0 authentication is enabled in Java.

In order to invoke a managed API with the OAuth 2.0 authentication method, API consumers must request an OAuth 2.0 token from the Informatica Intelligent Cloud Services OAuth 2.0 server.

You can use any OAuth 2.0 library, tool, or programming language to run the OAuth 2.0 authentication sequence. Before you run the OAuth 2.0 authentication, verify that you have the following information:

- URL of the Informatica Intelligent Cloud Services OAuth 2.0 server.
- OAuth 2.0 client ID and secret with permissions to run the managed API.

The following example shows the codes used for invoking a managed API with OAuth 2.0 authentication in Java:

```
import com.google.gson.Gson;
import com.squareup.okhttp.";

import java.io.IOException;
import java.util.Map;
import java.util.concurrent.Timeunit;

public class OAuthClientSample
(
public static String TEST_API_URL = "https://apigw-pod1.dm-us.informaticacloud.com/t/
apim.usw1.com/get_employee_details";
public static String OAUTH_SERVER_URL = "https://dm-us.informaticacloud.com/authz-
service/oauth/token";
public static String CLIENT_CREDENTIALS =
"YwliT1ZlMWJGRUpsOOhftenFFM8dZdjpRUjQzQXcwbes-";

OkHttpClient client = new OkHttpClient();

public static void main(String [] args) throws Exception
{
    new OAuthClientSample().runApi();
    }

    //
 //  run an OAuth 2.0 in a loop
    //
 public void runApi() throws Exception
{
    //
    //  obtain an OAuth 2.0 token for running the API
    //
 String token = getNewToken();
```

```
    while (true)

    //
    //  run the API using the OAuth 2.0 token
    //
    Request request = new Request.Builder()
    -url(TEST_API_URL)
    -method("GET", null)
    -addHeader("Authorization", Bearer + token)
    -build();
    Response response = client.newCall(request).execute();

    //
    // If the token expired, obtain a new token
    //
    if (response.code() --401)
                token = getNewToken ();
    else
        System.out.printIn(response.body().string());

        Thread.sleep(TimeUnit.SECONDS.toMillis(30));
        }
}

    /**
        * @return a new OAuth 2.0 token from the authentication server
        * @throws IOException
        */
        String getNewToken() throws IOException
        {
        String authHeader = "Basic " + CLIENT_CREDENTIALS;

        Request request = new request.Builder()
                -url(AUTH_SERVER_URL + "?grant_type =client_credentials")
                -method("POST", RequestBody.create(MediaType.parse("text/plain"), ""))
                -addHeader("Authorization", authHeader)
                -build();

        Response response = client.newCall(request).execute();

        if (response.code() != 200)
        {
                    System.err.printIn(response.code());
                    System.exit(1);
                    return null;
                }

        Map <String, Object> jsonResponse = new
Gson().fromJson(response.body().string(), Map.class);
        return (String)jsonResponse.get("access_token");

                }
}
```

Exporting deleted OAuth 2.0 clients

You can export deleted OAuth 2.0 clients for tracking purposes.

1. On the **Policies** page, click the **Authorization** tab.
2. Click **Export Deleted Clients**.

   API Manager downloads a `.csv` file that shows details of the OAuth 2.0 clients that were deleted since the time the organization was created. The file shows the client ID, client name, created time, last accessed time, and deletion time for all the deleted OAuth 2.0 clients.

JSON web token authentication

You can enable JSON Web Token (JWT) authentication for a managed Informatica Cloud Application Integration API that meets all of the following criteria:

- The associated process uses HTTP/SOAP binding.
- The associated process uses basic authentication and defines the user groups and users who can access the process service URL at run time.
- The associated process is published and exposed as a service.

JWT is an open standard that helps in the secure transmission of information between API consumers and REST web services such as Informatica Cloud Application Integration service APIs.

When you configure JWT authentication, you can generate a token using API Manager or API Portal and use the generated token to invoke the API. API consumers invoke the API by passing the token as a bearer token in the HTTP Authorization header.

An API token identifies an API by its name and version. If you delete an API and then create an API with the same name, you can continue to use the same token to invoke the API.

You can create groups of managed APIs and then generate a token for the group to use when invoking any JWT authenticated API in the group. You can add or remove APIs from the group.

A group token identifies an API group by its group ID. If you delete a group and then create a group with the same name, you can't continue to use the same group token to invoke APIs in the group. You must create a new token for the group.

If an API that has a token is part of a group that has a group token, you can use either token to invoke the API.

JWT tokens that you create for a managed API apply to the API version for which they are created. When you create a new version of an API where JWT authentication is enabled, generate JWT tokens for it.

JSON Web Token Authentication Tasks

After you publish an Informatica Cloud Application Integration process, Informatica Cloud Application Integration automatically exposes the service API to API Manager. You can then create a managed API for the Informatica Cloud Application Integration service API and make it available in API Portal.

Based on the role and privileges you are assigned, you can use API Manager or API Portal to perform JWT authentication tasks.

JWT Authentication Tasks in API Manager

Use API Manager to perform the following tasks for JWT authentication:

1. Configure JWT authentication for the managed API.
2. Generate a token and set an expiration date for the token. You can generate tokens for up to 15 APIs simultaneously. Optionally, you can make the managed API available in API Portal so that API Portal users can discover available APIs and view their authentication method.
3. Invoke the managed API by using the generated token.

JWT Authentication Tasks in API Portal

Use API Portal to perform the following tasks for JWT authentication:

1. View a list of managed APIs available in API Portal and view their authentication method.

2. Generate a token and set an expiration date for the token. You can generate tokens for up to 15 APIs simultaneously.
3. Invoke the managed API by using the generated token.

For more information about the JWT authentication tasks you can perform in API Portal, see the *API Portal* help.

JSON Web Token Expiration

API Manager uses the Coordinated Universal Time (UTC) time zone for the JWT token expiration and uses the current time on your computer as the baseline time for the token expiration. The token expires on the expiration date you configure and a minute earlier than the time at which you generated the token.

For example, if you generate the token on January 10 at 2:30 p.m. and set the expiration date as January 11, the token expires on January 11 at 2:29 p.m. If you set the expiration date as January 15, the token expires on January 15 at 2:29 p.m. The maximum expiration date for a token is 180 days from the current date.

After a token expires, you cannot refresh it. You must generate a new token.

Prerequisites

Before you configure JWT authentication in API Manager, you must perform the following prerequisite tasks in Informatica Cloud Application Integration:

1. Create a process and enable HTTP/SOAP binding for the process.
2. Configure basic authentication for the process by defining the user groups and users who can access the process service URL at run time.

   Note: You can configure JSON web token authentication for a managed API only if the associated process uses basic authentication in Informatica Cloud Application Integration. You cannot configure JWT authentication if the associated process allows anonymous access.
3. Publish the process to expose it as a service.

Configuring JSON Web Token authentication

After you create a managed API for a service that you published in Informatica Cloud Application Integration, you can configure JWT authentication, generate a token, and set an expiration date for the token. Optionally, you can make the managed API available in API Portal so that API Portal users can discover it in API Portal and invoke it.

1. On the **API Registry** page, click the managed API for which you want to configure JWT authentication.
2. On the **General** tab, from the **Authentication Method** list, select **JWT - JSON Web Token**.

   The **Generate JWT Access Token** area appears on the page.
3. Select an expiration date for the token and click **Generate**.

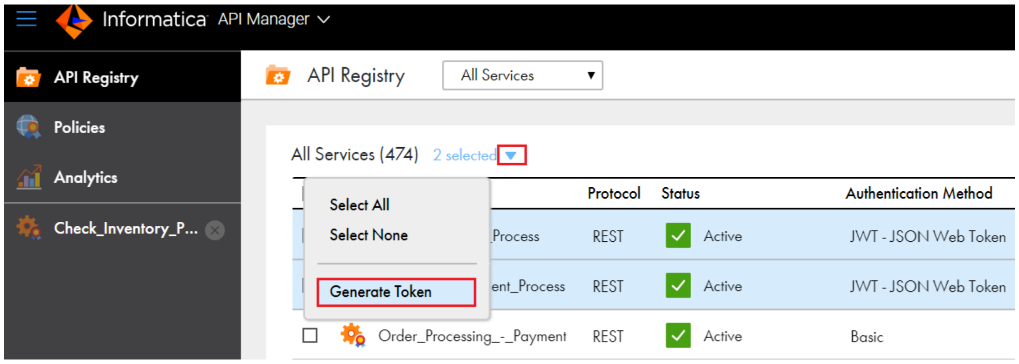
   API Manager creates a token for the managed API. The token appears on the page.

   Note: After you generate a token for the first time, the **Generate New Token** button appears. You can click this button to generate a new token if your earlier token has expired. After you generate a token, you cannot revoke the token.
4. Click **Copy Token** to copy the token and send the token to API consumers.

Generating JSON web tokens for multiple managed APIs simultaneously

You can generate tokens simultaneously for up to 15 managed APIs that are configured to use JSON Web Token authentication.

1. On the **API Registry** page, select the managed APIs to generate tokens for.
2. Click the down arrow above the list of APIs and select **Generate Token** as shown in the following image:

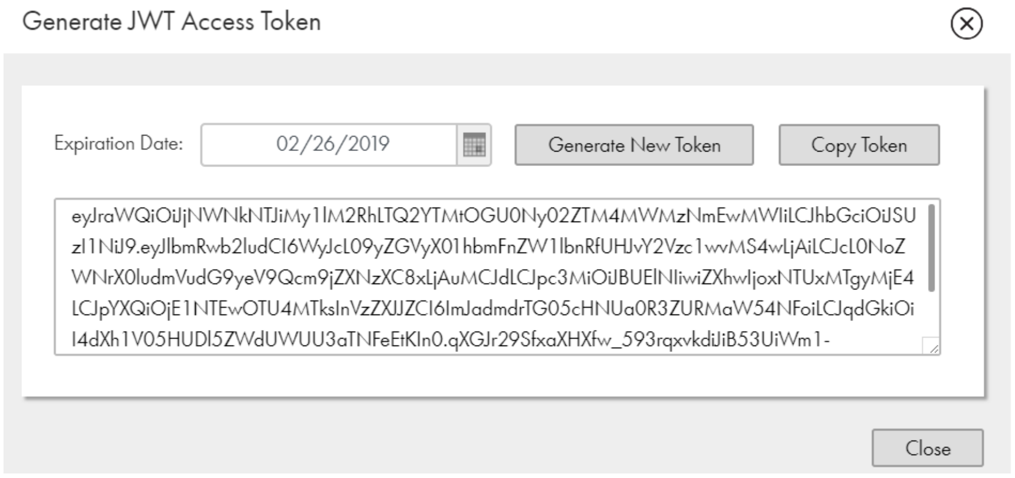


	Protocol	Status	Authentication Method
Process	REST	✓ Active	JWT - JSON Web Token
ent_Process	REST	✓ Active	JWT - JSON Web Token
Order_Processing_-_Payment	REST	✓ Active	Basic

The **Generate JWT Access Token** dialog box appears.

3. Select an expiration date for the token.
4. Click **Generate** to generate a token.

API Manager creates a token for the selected APIs as shown in the following image:



Expiration Date: 02/26/2019

eyJraWQiOiJjNWNkNTJiMy1lM2RhLTQ2YTMtOGU0Ny02ZTM4MWMzNmEwMWIiLCJhbGciOiJSU
zI1NiJ9.eyJlbmRwb2ludCI6WyJcL09yZGVyX01hbmFnZW1lbnRfUHJvY2Vzc1wvMS4wLjAiLCJcL0NoZ
WNrX0ludmVudG9yeV9Qcm9jZXNzXC8xLjAuMCJdLCJpc3MiOiJBUElNIiwiZXhwIjoxNTUxMTgyMjE4
LCJpYXQiOjE1NTEwOTU4MTksInVzZXJJZCI6ImJadmdrTG05cHNUa0R3ZURMaW54NFoiLCJqdGkiOi
I4dXh1V05HUDl5ZWdUWUU3aTNFeEtKIn0.qXGJr29SfxaXHXfw_593rqxvkdiJiB53UiWm1-

Note: After you generate a token for the first time, the **Generate New Token** button appears. You can click this button to generate a new token if your earlier token has expired. After you generate a token, you cannot revoke the token.

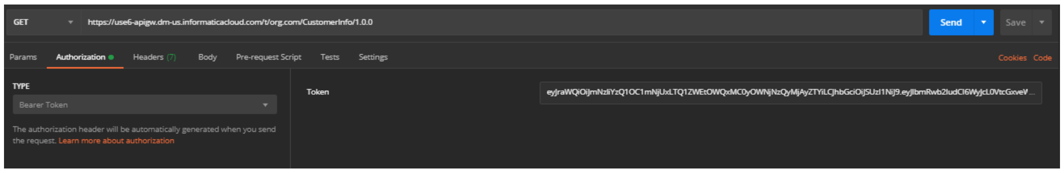
5. Click **Copy Token** to copy the token.

You can then invoke the API based on the authentication method it uses.

How API consumers invoke an API with JSON Web Token authentication

To invoke a managed API where JWT authentication is enabled, API consumers pass the token as a bearer token in the HTTP Authorization header.

The following image shows an API invoked through Postman with a Bearer Token authorization type and the token that the API consumer entered:



https://use6-apigw.dm-us.informaticacloud.com/t/org.com/CustomerInfo/1.0.0

TYPE: Bearer Token

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

Token: eyJraWQiOiJmNzliYzQ1OC1mNjUxLTQ1ZWEtOWQxMC0yOWNjNzQyMjAyZTYiLCJhbGciOiJSUzI1NiJ9.eyJlbmRwb2ludCI6WyJcL0VtcGxveV...

CHAPTER 7

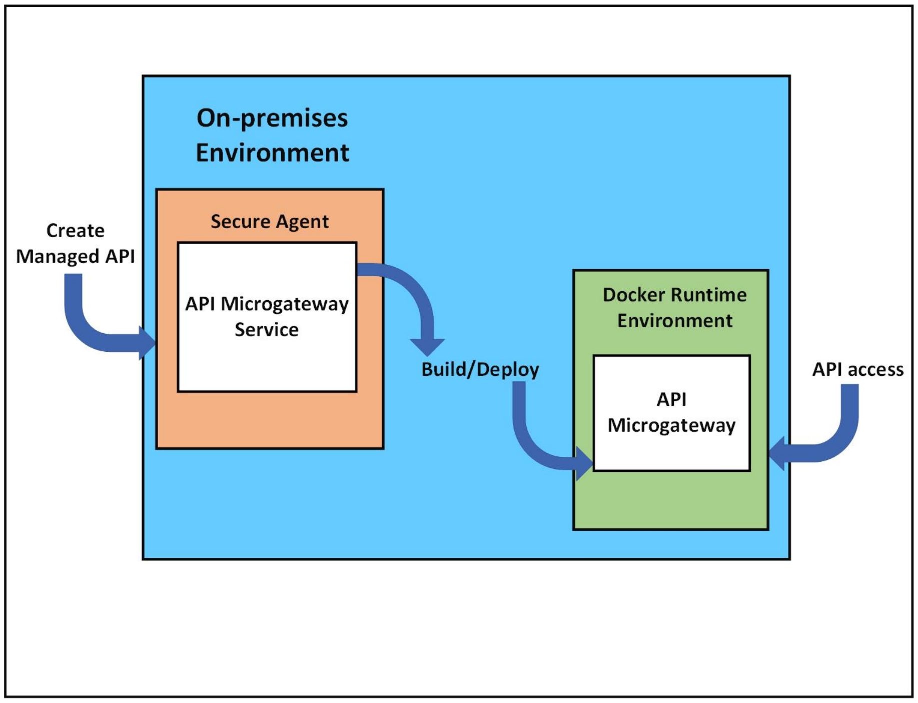
API Microgateway Service

The API Microgateway Service manages Application Integration processes that run on the organization's on-premises Secure Agent. Use the API Microgateway Service to expose managed APIs as API Microgateway proxies.

The API Microgateway Service provides REST APIs to create and deploy API Microgateway proxies. API consumers access the managed APIs deployed as API Microgateway proxies on the organization's on-premises environment. The Application Integration processes expose REST Service URL and SOAP Service URL endpoints.

Use the API Microgateway Service to build an API Microgateway proxy to an API endpoint to manage. The API Microgateway Service builds an API Microgateway as an immutable Docker image on the organization's Secure Agent machine. You then use the API Microgateway Service to deploy the Docker image in a container on the Secure Agent Docker runtime environment for API access. The API Microgateway applies the API access policies that you configure before forwarding the requests to Application Integration endpoints.

The following diagram shows the API Microgateway Service and API Microgateway components exposing a managed API in the on-premises environment:



The Secure Agent Docker runtime environment hosts the Docker images using the blue-green deployment strategy to provide zero downtime during updates of the API Microgateway component.

Access control of managed APIs that you expose with the API Microgateway Service

You can configure the following methods to control access to managed APIs that you expose with the API Microgateway Service:

- IP filtering policy. Specify multiple IP rules to grant or deny access to various IP ranges. The IP filtering order of precedence is based on the listing of the IP rules that you specify. IP rules are overridden based on order. For example, setting `DENY` and `ALLOW` as default IP rules does not indicate that all functionalities will be both allowed and denied.

  Note: Always enter default IP rules as the first rules on the list.
- Rate limit policy. The rate limit policy places usage restrictions on each managed API that defines the allowed limit of usage for a designated time period.
- Authentication. Enable OAuth 2.0 authentication for the API Microgateway Service and use it with managed APIs that invoke an Application Integration process that uses basic authentication. OAuth 2.0 authentication verifies user access credentials with Informatica's Identity Service.

Expose a managed API with the API Microgateway Service

Use the API Microgateway Service to expose a managed API on the organization's on-premises environment.

Before you expose a managed API, perform the prerequisite tasks on your organization's Secure Agent machine. After you complete the prerequisites, follow the steps to create, build, and deploy an API Microgateway proxy.

After you expose a managed API, if you make changes to the on-premises environment, you might need to make one or both of the following changes:

- Edit the API Microgateway Service properties in Administrator. For example, you might want to change the name of the project that stores the API configurations when you create a new project.
- Enable the API Microgateway Service for a Secure Agent or a Secure Agent group in Administrator. For example, you might want to change the Secure Agent that runs the API Microgateway Service when you create a new project.

For more information about editing the API Microgateway Service properties or enabling the API Microgateway Service for a Secure Agent or a Secure Agent group, see *Secure Agent Services* in the Administrator help.

Prerequisites for exposing a managed API with the API Microgateway Service

Before you expose a managed API with the API Microgateway Service, perform the following tasks on the organization's Secure Agent machine:

- Configure the Docker runtime environment on the same Secure Agent that runs the API Microgateway Service. Certified Docker runtime environments include Linux Docker Engines and Docker for Windows.

- Enable TCP without TLS to run Docker daemon.
- Run the API Microgateway Service on a Secure Agent with TLS version 1.2 or a later enabled and versions TLS 1.0 and 1.1 disabled. For more information, see HOW TO: Disable TLS 1.0 and 1.1 on Secure Agent.
- Generate a new SSL certificate and PEM file.

Generating SSL certificate for the API Microgateway

Perform the following steps to generate a new SSL certificate and PEM file for the API Microgateway:

1. Install OpenSSL on the organization's Secure Agent machine, if it is not already installed. Open OpenSSL.
2. Generate an RSA key (size: 2048). Output the RSA key to a file named `mydomain.key`, using the command: `openssl genrsa -out mydomain.key 2048`

   If you change the `mydomain.key` file name, you must also change the `haproxy.cfg` file in the Agent Data folder, `data\apimgw_docker\haproxy`, to reflect the corresponding file name.
3. Generate a certificate signing request using the command: `openssl req -new -key mydomain.key -out mydomain.csr`
4. Generate a signed certificate using the command: `openssl x509 -req -days 730 -in mydomain.csr -signkey mydomain.key -out mydomain.crt`
5. Create a certificate package using the command: `bash -c 'cat mydomain.key mydomain.crt >> <dir>/mydomain.pem'`
6. When you receive the SSL certificate, copy and paste the contents of the certificate file into the PEM file containing the RSA key. The certificate file name must be `mydomain.pem`

Creating a managed API to expose with the API Microgateway Service

Create a managed API to expose with the API Microgateway Service to use as an API Microgateway proxy to an API endpoint to manage.

1. Access the Application Integration service and then click **Create a Process**.

   The **Process** window opens.
2. In the **Process Properties** panel, click **Start**. Enter the required properties and then click **Save**.
3. Click **Publish** to publish the process on the Secure Agent.
4. Click **Properties Detail**. In the Endpoints area, click **Copy** to copy the Service URL or the SOAP Service URL and then click **Close**.
5. Open the following URL template:

   ```
   https://{<host>}:{<port number>}/api/v1/agent/apimgw/apispec
   ```
   Where:
   - `<host>` is the machine that hosts the Secure Agent that runs the API Microgateway Service.
   - `<port number>` is the number of the port that the API Microgateway Service is assigned during startup. You can find the port number in the following file: `<Secure Agent installation directory>/apps/ApiMicrogatewayService/logs folder/apimgw.log`
6. Paste the URL you copied to the path element of the URL template.

   API manager assigns the managed API name as the value of the id element.

7. Optionally, define endpoint access policies for the managed API. You can define a rate limit policy, an authentication type, and an IP filtering policy for the managed API.

   a. Define the rate limit policy. Enter one of the following values for timeUnit:

      min

      hour

      day

      week

      month

      year

   b. Define the authentication type. If you use OAuth 2.0 authentication, enter the access token that you generate in Application Integration in the request header when you create the managed API. If no value is defined, the authentication type is anonymous. Enter one of the following values for authType:

      BASIC

      NO_AUTH

      OAUTH2

   c. Define as many IP range rules as needed. Enter the rule type and the IP address range for each ipRangeRules section. You can enter one of the following values for type:

      ALLOW

      DENY

   The following example shows a payload of a request to create a managed API with a rate limit policy, basic authentication, and an IP range rule that allows all URLs:

```
{
  "id": "getbasic",
  "path": "/rt/BasicAuth_Lin",
  "methods": [
    "GET",
    "POST"
  ],
  "policies": {
    "rateLimit": {
      "limit": 4,
      "duration": 1,
      "timeUnit": "min"
    },
    "authType": "BASIC",
    "ipRangeRules": [
      {
        "type": "ALLOW",
        "high": "255.255.255.255",
        "low": "0.0.0.0"
      }
    ]
  },
  "endpoints": [
    "https://{{endpointhost}}:{endpointport}/process-engine"
  ]
}
```

Building an API Microgateway

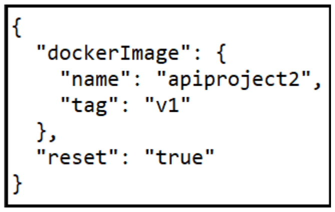
Use the API Microgateway Service to build an API Microgateway as an immutable Docker image for each managed API:

▶ Build the API Microgateway to an API endpoint using the URL:

```
https://{<host>}:{<port number>}/api/v1/agent/apimgw/build
```

The Docker image name must not contain the following characters: - _ , . If the Docker image name includes restricted characters, the build fails.

The API Microgateway Service builds the API Microgateway as a Docker image on the repository of the Docker runtime environment. The following image shows an example payload of invoking a POST call to build an API Microgateway:



```
{
  "dockerImage": {
    "name": "apiproject2",
    "tag": "v1"
  },
  "reset": "true"
}
```

Deploying an API Microgateway

Use the API Microgateway Service to deploy an API Microgateway in a Docker image container on the organization's Secure Agent Docker runtime environment:

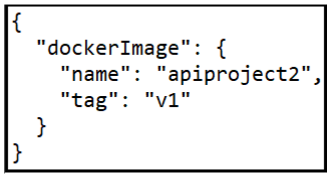
▶ Deploy the API Microgateway using the URL:

```
https://{<host>}:{<port number>}/api/v1/agent/apimgw/deploy
```

Access the managed APIs that are deployed and running on the organization's Secure Agent using the URL:

```
https://{<host>}:{<port number>}/apimgw/{path}
```

The following image shows an example payload of invoking a POST call to deploy an API Microgateway in a Docker image container:



```
{
  "dockerImage": {
    "name": "apiproject2",
    "tag": "v1"
  }
}
```

OAuth 2.0 authentication for the API Microgateway Service

You can enable OAuth 2.0 authentication for the API Microgateway Service, to use with managed APIs that invoke an Application Integration process that uses basic authentication.

Use an application or service that supports token authentication, such as Postman, to configure an OAuth 2.0 client and to enable OAuth 2.0 authentication for the API Microgateway Service.

You can use basic authentication or a bearer token to configure OAuth 2.0 authentication for the API Microgateway Service. If you use basic authentication, enter an Informatica Intelligent Cloud Services user name and password in the authorization header. If you use a bearer token, enter the access token of the managed API in the authorization header. For more information, see Chapter 6, "Authentication and authorization" on page 38.

Client tokens that you use for OAuth 2.0 authentication for the API Microgateway Service time out after a defined timeout period. After a token times out, you can't use it. You must regenerate the token.

The default timeout is 60 minutes. You can set a different timeout when you create the OAuth 2.0 client or change it later.

Use APIs to perform the following tasks for the API Microgateway Service:

- List all OAuth 2.0 clients.
- View, delete, download, or edit an OAuth 2.0 client using the client ID.
- Regenerate an OAuth 2.0 client secret.
- Disable or enable an OAuth 2.0 client.
- Download a list of deleted OAuth 2.0 clients.
- Get the access token for an OAuth 2.0 client.

Create an OAuth 2.0 client for the API Microgateway Service

Create an OAuth 2.0 client that enables managed API consumers to invoke managed APIs for the API Microgateway Service where OAuth 2.0 authentication is enabled by using an application or service that supports token authentication.

1. Open the following URL template:

   `https://{<host_URL>}/apimgmt/v0.4/oauth/client`
   where, `<host_URL>` is the API Manager POD URL.

   For example, `apim-pod2.dm-us.informaticacloud.com` for US East POD.
2. Edit the values of the request body to enable the OAuth 2.0 client for API Microgateway.
   a. In the `"name"` field, enter a name for the OAuth 2.0 client. The name is case sensitive and must be unique in the organization.

      The name can contain up to 32 characters, including any letter on the ASCII table, non-Latin characters, any digit, spaces, and the special characters . _ and -. The name cannot contain any of the following characters: / " ' < > & : ~ ! @ # ; % ^ * + = { } | , \
   b. Optionally, in the `"description"` field, enter a description of the OAuth 2.0 client.

      The description can contain up to 1,024 characters, including any letter on the ASCII table, non-Latin characters, any digit, spaces, and the special characters . _ and -.
   c. Optionally, in the `"accessTokenTimeout"` field, enter a timeout value in minutes for the access token.

      You can enter a value from 5 through 1440. If you don't enter a timeout value, the token timeout is 60 minutes.
3. Invoke a POST call.

   The OAuth 2.0 client is enabled for all the API Microgateway Service managed APIs in the organization that invoke an Application Integration process that uses basic authentication. You can view the enabled OAuth 2.0 client on the **OAuth 2.0 Clients** table on the **Authorization** tab of the **Policies** page in API Manager.
4. On the **Authorization** tab, click **Copy URL** and send the access token to API Microgateway consumers.

Manage OAuth 2.0 clients for the API Microgateway Service

Use APIs to manage OAuth 2.0 clients for the API Microgateway Service. Perform the following tasks to view, delete, and edit OAuth 2.0 clients, and to download a list of deleted clients:

View all OAuth 2.0 clients

Use the following URL to request a list of all the OAuth 2.0 clients for API Microgateway:

```
https://{<host_URL>}/apimgmt/v0.4/oauth/client/
```

Invoke a GET call to receive a list of all the OAuth 2.0 clients.

Download a list of deleted OAuth 2.0 clients

Use the following URL to download a list of the deleted OAuth 2.0 clients for API Microgateway:

```
https://{<host_URL>}/apimgmt/v0.4/oauth/client/download/deleted-clients
```

Invoke a GET call to download the list in JSON format.

View, delete, and edit OAuth 2.0 clients

Use the following URL to view, delete, or edit an OAuth 2.0 client for API Microgateway:

```
https://{<host_URL>}/apimgmt/v0.4/oauth/client/{<clientID>}
```

Invoke a GET call to view the OAuth 2.0 client details.

Invoke a DEL call to delete the OAuth 2.0 client.

Invoke a PATCH call to edit the OAuth 2.0 client details. Edit the values of the request body to update the OAuth 2.0 client.

Regenerate an OAuth 2.0 client secret for the API Microgateway Service

You can regenerate an OAuth 2.0 client secret for the API Microgateway Service if needed. When you regenerate the client secret, API Manager disables the current client secret.

1. Open the following URL template:

   ```
   https://{<host_URL>}/apimgmt/v0.4/oauth/client{<clientID>}newSecret
   ```
2. Paste the client ID of the OAuth 2.0 client into the URL template.
3. Invoke a GET call.

   API Manager regenerates the client secret and authorization header value.
4. On the response body, copy the new client secret and send it to API Microgateway consumers.

Disable and enable OAuth 2.0 clients for the API Microgateway Service

When you create an OAuth 2.0 client for the API Microgateway Service, it is enabled by default. You can disable a client and enable a disabled client. API consumers can't use disabled OAuth 2.0 clients for authentication.

1. Open the following URL template:

   ```
   https://{<host_URL>}/apimgmt/v0.4/oauth/client/{<clientID>}/{state}
   ```
   Where `<host>` is the machine that hosts the Secure Agent that runs the API Microgateway Service.
2. In the `"state"` field, enter `"disable"`.

3. Invoke a POST call.

   The response body shows the state as "`DISABLED`".
4. To enable a disabled Auth 2.0 client, in the "`state`" field, enter "`enable`".
5. Invoke a POST call.

   The response body shows the state as "`ENABLED`".

Get the OAuth 2.0 client access token for the API Microgateway Service

Get the OAuth 2.0 client access token for the API Microgateway Service. Default timeout value is 15 minutes for the access token.

1. Open the following URL template:

   `https://{<host_URL>}/apimgmt/v0.4/oauth/client/accessToken`
2. Paste the client ID and client secret of the OAuth 2.0 client into the request body.
3. Invoke a POST call.

   API Manager generates the OAuth 2.0 client access token.
4. On the response body, copy the client access token and send it to API Microgateway consumers.

Running a managed API that you expose with the API Microgateway Service

Run a managed API that you expose with the API Microgateway Service.

1. Open the following URL template:

   `https://{<host>}:{<port number>}/apimgw/<path>`

   Where:
   - `<host>` is the machine that hosts the Secure Agent that runs the API Microgateway Service.
   - `<port number>` is the number of the port that the API Microgateway Service is assigned during startup. You can find the port number in the following file: `<Secure Agent installation directory>/apps/ApiMicrogatewayService/logs folder/apimgw.log`
   - `<path>` is the URL that you copied when you created the managed AP.
2. Define the type of authentication:
   - Anonymous. Select "No Auth" in the authorization header.
   - OAuth 2.0. You can use basic authentication or a bearer token to configure OAuth 2.0 authentication. To use basic authentication, select "Basic" in the authorization header and enter an Informatica Intelligent Cloud Services user name and password. To use a bearer token, select "Bearer Token" in the authorization header and paste the access token of the managed API in the `token` field.
3. Invoke a GET call to run the managed API.

   API Manager runs the managed API on the Docker runtime environment.

View all managed APIs for the API Microgateway Service

Request a list of all the managed APIs that you create with the API Microgateway Service on a Secure Agent using the following URL:

```
https://{<host>}:{<port number>}/api/v1/agent/apimgw/apispec
```

Invoke a GET call to receive a list of all the managed APIs.

View, delete, and edit a managed API for the API Microgateway Service

View or delete a managed API that you create with the API Microgateway on a Secure Agent, or edit the API endpoint policies of a managed API that you create with the API Microgateway on a Secure Agent by opening the following URL:

```
https://{<host>}:{<port number>}/api/v1/agent/apimgw/apispec?path={<path>}
```

Invoke a GET call to view the managed API.

Invoke a DEL call to delete the managed API.

Edit the values of the page to update the endpoint access policies of the managed API.

Access and event logs

The API Microgateway Service stores API usage audits of the blue and green Docker image containers in access and event logs.

The logs are available in the API Microgateway Service logs folder on the organization's Secure Agent. The logs can be found in the following file path:

```
<Secure Agent installation directory>/apps/ApiMicrogatewayService/logs
```

Troubleshooting the API Microgateway Service and API Microgateway

Use the following sections to troubleshoot errors in the API Microgateway Service and API Microgateway.

Troubleshooting API Microgateway Service

I received the custom license for the API Microgateway Service from Informatica Global Customer Support, but the service is not available on the Secure Agent.

View the Secure Agent core logs to check if there were any issues with package download and installation on the Secure Agent. Contact Informatica Global Customer Support in case of errors.

The API Microgateway Service enters stopped state, error state, or restarts.

Check the Secure Agent core and audit logs for specific symptoms that could reveal the cause of error. Low memory resources of the Secure Agent host can also interrupt the API Microgateway Service.

I can't find the secure channel port of the API Microgateway Service.

The Secure Agent core platform assigns a secure channel port to the API Microgateway Service during startup. The port is changed during the service restart. You can find the port information in the following file: `<Secure Agent installation directory>/apps/ApiMicrogatewayService/logs folder/apimgw.log`

The API Microgateway build failed.

There are two types of errors that can cause the API Microgateway build to fail:

1. WSO2 errors can occur if the managed API and its configured policies do not conform to the OpenAPI Specification. If the project enters an irreversible corrupt state, you must create a new project with a new managed API and configurations. To create a new project, change the project name and restart the Secure Agent. For more information, see *Secure Agent Services* in the Administrator help.
2. Docker image generation errors can occur if the Docker image or tag name includes the following restricted characters: `- _ , .`. These configurations are validated by the API Microgateway build, and failure will indicate possible causes.

The API Microgateway deploy failed.

API Microgateway deploy failures are primarily related to Docker image deployment. You will receive error messages that include the cause of failure.

In case of API Microgateway blue-green deployment, deployment will fail if the previous Docker image is removed from the registry. To avoid this issue, you can pass the reset flag in the request payload to force resetting the blue-green deployment states.

Troubleshooting API Microgateway

I am unable to access the managed APIs I deployed as API Microgateway proxies.

If you are unable to access the managed APIs you deployed, complete the following tasks:

- Verify that the blue and green Docker image containers are running.
- View the API Microgateway access and event logs in the file path: `<Secure Agent installation directory>/apps/ApiMicrogatewayService/logs`. Verify that the haproxy Docker image container is running and check the corresponding logs for any routing failures.

The rate limit policy I configured was not applied.

API Manager applies the configured rate limit policy according to the allowed usage in a designated time period. For example, if the rate limit is five requests per minute, then the rate limit counter will reset at the start of every minute. In addition, the rate limit counter is reset when the API Microgateway is updated and redeployed.

The IP filtering policy I configured was not applied.

API Manager applies the IP filtering policy according to the order of precedence and listing of IP rules that you specify. The first rule on the list should be the default IP rule, which can deny or allow the entire IP range. View the API Microgateway access and event logs to check if the IP rules were applied and their results.

I want to stop the API Microgateway.

To stop the API Microgateway, stop all three Docker image containers.

CHAPTER 8

Analytics

API analytics provide a graphical overview of activity and API usage, as well as the ability to drill down to specific activities and events. The Overview dashboard offers a collection of panels that contain reports about APIs. Use the dashboard to view visual summary information about APIs, such as trends in usage over time, APIs with the most invocations, and the most frequent users.

When users invoke API calls, the organization can track general API usage activity for API access instances and access exceptions. The organization may need to track access exceptions to accommodate business or legal needs. API Manager creates an activity log for all API access instances, and an event log to track any access exceptions that users create when invoking APIs.

The organization can create IP filtering, rate limiting, basic authentication, and privacy policies. If an API call breaches a policy, API Manager logs the incident in the event log.

Overview reports

You can use the **Overview** page in the **Analytics** page to view graphical summary information about APIs, including trends in usage over time, APIs with the most invocations, and the most frequent users.

The **Overview** page shows API usage trends for a selected period, for 7, 30, or 90 days. You can refresh the data by clicking the refresh icon. The last time that you refreshed the data is displayed near the icon.

Note: Data from the current day appears after a delay of half an hour.

You can also view the APIs that were most frequently invoked in the selected period, ranked by number of invocations. To sort the display, click the title of the column by which to sort the display.

You can view the users who most frequently invoked APIs in the selected period, ranked by number of invocations. To sort the display, click the title of the column by which to sort the display.

Overview report properties

The following table describes the properties of the Top APIs report in the **Overview** page:

Property	Description
API Name	Name of the API.
API URL	Identifies the URL of the API that was invoked.

Property	Description
Protocol	Identifies the protocol of the API: - REST - SOAP
Invocations	Number of times that the API was invoked during the selected period.

The following table describes the properties of the Top Users report in the **Overview** page:

Property	Description
Username	Name of the user who invoked the API, if known.
Invocations	Number of times that the user invoked URLs for APIs during the selected period.

Activity log

You can use the **Activity Log** tab in the **Analytics** page to view API access requests for a selected date range.

The **Activity Log** tab shows API access attempts in chronological order. To sort the access logs, click the title of the column to sort. The last time that you refreshed the data is displayed beside the **Find** field.

Note: The timestamp displayed is based on the local time zone setting of your browser.

Activity log properties

The following table describes the **Activity Log** tab properties:

Property	Description
Timestamp	Time that the access occurred. The timestamp displayed is based on the local time zone settings of your browser.
API Name	Name of the API.
API Version	Version of the API.
API URL	Identifies the URL of the API that was invoked.
Protocol	Identifies the protocol of the API: - REST - SOAP
Method	Identifies the API call method.
Authentication	Identifies the authentication type.
HTTP Response	The HTTP response to the API invocation.

Property	Description
Consumer	Name of the user who initiated the API call. For managed APIs that use OAuth 2.0 authentication, consumer details also include the ID of the OAuth 2.0 client that invoked the API.
IP Address	IP address that accessed the API.
Duration	The duration of access measured from the moment the API request reaches API Manager until the moment API Manager provides a response. If you enable response caching for an API, when the response arrives from the cache, (**Cached**) appears next to the duration.

Downloading activity logs

Download API activity logs from the **Analytics** page. You can download up to 500 log entries as a CSV file for a date range or download a ZIP file of the logs for a specific month from the last six months.

1. Navigate to the **Analytics** page. On the **Activity Log** tab, perform one of the following actions:
    - To download logs for a date range, select dates from the date range menu and click **Show Log**. The **API Invocations** table shows logs for the date range.
    - To download logs for a specific month, select a month from the **Download log (ZIP)** list.
2. Click **Download**.

    Note: The ZIP file includes one or more current log files. API Manager renames the current files according to the date, and creates current files for new activity log entries once per day. During new releases, API Manager doesn't rename the existing current files, but creates new current files.

Event log

You can use the **Event Log** tab in the **Analytics** page to view policy breaches on APIs for a selected date range.

The **Event Log** tab shows the logged incidents in chronological order. To sort the incidents based on a different criterion, click the title of the column by which to sort the display. The last time that you refreshed the data is displayed beside the **Find** field.

Note: The timestamp displayed is based on the local time zone setting of your browser.

Event log properties

The following table describes the **Event Log** tab properties:

Property	Description
Timestamp	Time that the access exception or privacy policy leakage occurred. The timestamp displayed is based on the local time zone of your browser.
API URL	URL of the API that was invoked.

Property	Description
Authentication	Identifies the authentication type.
HTTP Response	HTTP response to the API invocation.
Description	Description of the access exception or privacy policy leakage, and the action that API Manager took, if any.
Consumer	Name of the user who initiated the API call. For managed APIs that use OAuth 2.0 authentication, consumer details also include the ID of the OAuth 2.0 client that invoked the API.
IP Address	IP address that accessed the API.

Downloading event logs

Download API event logs from the **Analytics** page. You can download up to 500 log entries as a CSV file for a date range or download a ZIP file of the logs for a specific month from the last six months.

1. Navigate to the **Analytics** page. On the **Event Log** tab, perform one of the following actions:
    - To download logs for a date range, select dates from the date range menu and click **Show Log**. The **Events** table shows logs for the date range.
    - To download logs for a specific month, select a month from the **Download log (ZIP)** list.
2. Click **Download**.

    Note: The ZIP file includes one or more current log files. API Manager renames the current files according to the date, and creates current files for new event log entries once per day. During new releases, API Manager doesn't rename the existing current files, but creates new current files.

Searching for a log

You can search for an event or activity log by date of creation or by searching for specific text in the display columns.

1. To search for logs that were created during a specific time period, in the relevant tab, select a range of dates in the **Select date range** fields, and then click **Show Log**. Ensure that you select dates based on the local time zone setting of your browser.

    Logs for the selected time period are displayed.
2. To sort the logs according to a specific property, click the column picker icon to the left of the **Find** field and then select the column by which to sort the logs.
3. To search for logs based on specific descriptive text, in the **Find** field, type the text for which to search.

    The log table shows the relevant logs.

INDEX