



User Guide | PUBLIC
2024-04-25

SAP API Management Standalone Service

Content

- 1 SAP API Management in the Cloud Foundry Environment. 4**
- 1.1 What is API Management. 5
 - Components of API Management. 8
 - Concepts of API Management. 10
 - API Services. 12
 - Accessibility Features in API Management. 13
 - Important Notes. 13
- 1.2 What's New for SAP API Management Cloud Foundry. 14
 - Archive - Release Notes for SAP API Management. 85
- 1.3 Patch Releases for API Management. 117
 - Patch Archive 2021. 118
- 1.4 Configure. 120
 - Initial Setup. 120
 - Custom Domain Configuration for API Portal or API business hub enterprise Subscription 162
 - Region-Specific IP Addresses Available for API Management Cloud Foundry Environment. 163
 - Configuring Additional Virtual Host in Cloud Foundry Environment. 165
 - Shadow Users. 176
 - Cancel API Management Service Subscription. 177
 - Centralized API business hub enterprise [Classic Design] 177
 - Centralized API business hub enterprise [New Design]. 186
- 1.5 Build API Proxies. 195
 - Key Components of an API. 196
 - Different Methods of Creating an API Proxy. 477
 - Additional Configurations. 537
- 1.6 Test API Proxies. 648
 - Debug an API Proxy. 650
- 1.7 Publish API Proxies 652
 - Create a Product. 653
 - View Applications. 660
- 1.8 Consume API Proxies 661
 - Onboard an Application Developer. 663
 - Configure the API business hub enterprise [Classic Design]. 671
 - Configure the API business hub enterprise [New Design]. 675
 - Customize the Visual Format of the API business hub enterprise 677
 - Manage Domain Categories [New Design]. 679

	Manage Notifications [New Design].	681
	Manage External Content (New Design).	682
	Manage Access	683
	Subscribe to a Product [Classic Design].	685
	Subscribe to a Product [New Design].	686
	Create an Application [Classic Design].	686
	View Applications, Costs, and Analyze Reports [New Design].	693
	Create an Application [New Design].	693
	Consume Applications.	697
	Analyze Applications.	698
	Consume API Proxies Using SAP Business Application Studio.	699
	Test Runtime Behavior of APIs [New Design].	699
1.9	Analyze APIs.	701
	API Analytics.	702
	Advanced API Analytics.	706
	SAP Analytics Cloud for	717
1.10	Monetize APIs.	717
	Rate Plan Service.	718
	Billing Service	723
	Create or Update or Read an Application using Subscription key.	727
1.11	Discover API Packages.	731
	Package Details.	732
1.12	API Documentation.	732
1.13	Security.	733
	Data Protection and Privacy for API Management.	734
1.14	Monitoring and Troubleshooting.	776
	Limits in API Management.	776
	Monitor the Health of Custom Domain Virtual Host Certificates Using SAP Cloud ALM.	780
1.15	Migration of API Management Content.	780
	Migrating API Management from Neo to Cloud Foundry Environment.	781
	Migration of API Management Content between Cloud Foundry Environments.	817

1 SAP API Management in the Cloud Foundry Environment

SAP API Management lets you publish, promote, and oversee APIs in a secure and scalable environment.

Environment

This service runs in the following environments:

- NEO environment
- Cloud Foundry environment

Features

Create omni-channel experiences	Use API Designer and Open APIs to create a omni-channel mobile experience across devices.
Secure your digital assets, interfaces	Help protect your data and digital assets in this hyper-connected world. Get deep insights on API usage.
Manage the end-to-end lifecycle of APIs	Scale billions of API calls to unlock new opportunities, new business potential and add additional value.
Engage developers and partners	API Business Hub Enterprise simplifies sharing managed APIs and collaborations with customers, partners, and developers.
Grow new revenue streams	Monetize your data and digital assets with help of API Portal. Upsell and cross-sell through your ecosystem.
Evolve B2B integrations	Extend solutions with additional SAP BTP capabilities for mobile, offline and integration.
Benefit from multitenancy support	Use this service in tenant-aware plications.

API Management technology helps you to share digital assets and enable developer communities to consume these assets in new channels, devices, and user interfaces. Available in the cloud, the technology helps promote coinnovation among employees, partners, and the developer community. To gain better insights about consumer needs, you can empower employees and partners with access to critical information and increase reach to a wider customer base.

The Cloud Foundry environment gives you the ability to subscribe to the API Management service, while you may choose a public infrastructure to run the API Management service, such as Amazon Web Services or microsoft Azure.

Get started by subscribing to API portal and API business hub enterprise applications where you can create APIs and consume them. For setting up the API Portal application, see [here \[page 120\]](#). Once the API portal is setup, see [here \[page 125\]](#) to set up the API business hub enterprise application.

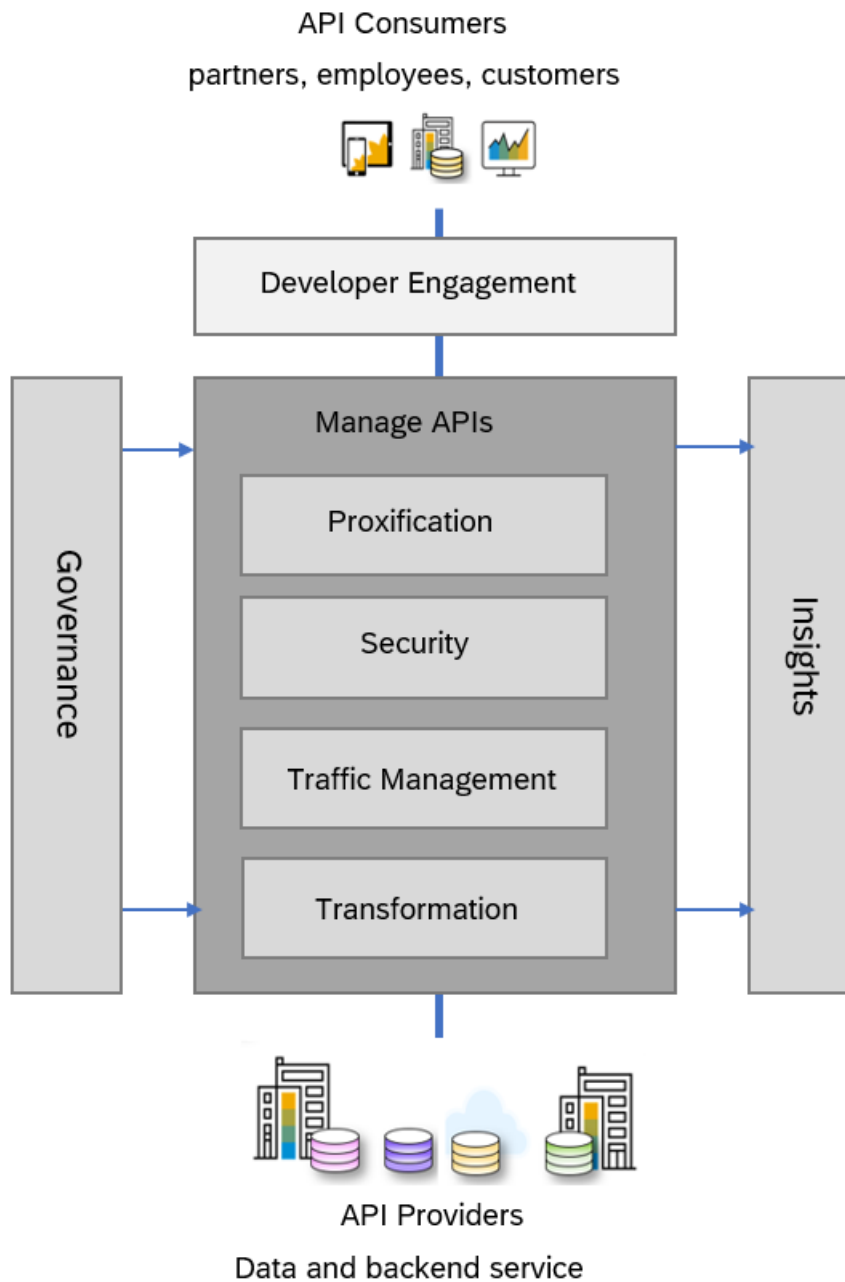
1.1 What is API Management

API Management lets you publish, promote, and oversee APIs in a secure and scalable environment. Using API Management, you can create simple digital experiences for your consumers, partners, and employees.

The API Management capability in SAP Integration Suite is a complete solution, that addresses all enterprise requirements for API security and governance.

With API Management you can:

- **Proxify your APIs:** Create your own unified and harmonised API presence, using your own domain.
- **Secure your APIs:** Secure your APIs against unauthorized access and threats. API management helps organizations define a standardized set of policies to protect APIs and the underlying backends.
- **Perform Traffic Management:** Configure cache, and control traffic quotas and spikes, using the traffic management policies.
- **Govern your APIs:** Discover and document all your APIs, manage the lifecycle of your APIs and govern them using the policies. Over 30 different policy types are available, ranging from traffic management and security policies.
- **Get Business Insights:** Monitor with usage analytics, logs, events and triggers; use business insights to monetize your APIs.
- **Transform your APIs:** Apply advance header and payload modifications.
- **Developer Engagements:** API business hub enterprise is a feature-rich, themed, and customizable portal designed specifically for application developers. It provides comprehensive API documentation, code snippets, and more. With API business hub enterprise, developers can easily engage with the platform, enabling them to discover, subscribe to, and consume APIs directly.



Features

Create omni-channel experiences

Use API Designer and Open APIs to create a omni-channel mobile experience across devices.

Secure your digital assets, interfaces

Help protect your data and digital assets in this hyper-connected world. Get deep insights on API usage.

Manage the end-to-end lifecycle of APIs	Scale billions of API calls to unlock new opportunities, new business potential and add additional value.
Engage developers and partners	API Business Hub Enterprise simplifies sharing managed APIs and collaborations with customers, partners, and developers.
Grow new revenue streams	Monetize your data and digital assets with help of API Portal. Upsell and cross-sell through your ecosystem.
Evolve B2B integrations	Extend solutions with additional SAP BTP capabilities for mobile, offline and integration.
Benefit from multitenancy support	Use this service in tenant-aware applications.

Use Cases

With the emergence of cloud, mobile and social technologies, new applications have become a driving force in the way people consume content and access services. Millions of mobile devices in use today are generating digital data at an exponential rate. This massive influx of digital information is changing the way businesses are operating. To keep up with the digital footprint produced, businesses identify and implement ways to reach out to their customer and meet their needs, easily and securely.

Through software interfaces called application programming interfaces (APIs), companies can provide business services and information directly to customers. APIs simplify the work of programming graphical user interface components for all types of apps on mobile devices, in the cloud, and on wearables. Exposing digital assets enable you to create and deliver content and business services to your customers, partners, and employees. That way they can better engage, collaborate, and innovate.

API Management technology helps you to share digital assets and enable developer communities to consume these assets in new channels, devices, and user interfaces. Available in the cloud, the technology helps promote coinnovation among employees, partners, and the developer community. To gain better insights about consumer needs, you can empower employees and partners with access to critical information and increase reach to a wider customer base.

API Management facilitates consumer engagement anywhere, any time. It reduces complexity by leveraging a single provisioning platform (API Platform) to provide unified access and governance of APIs across a heterogeneous landscape.

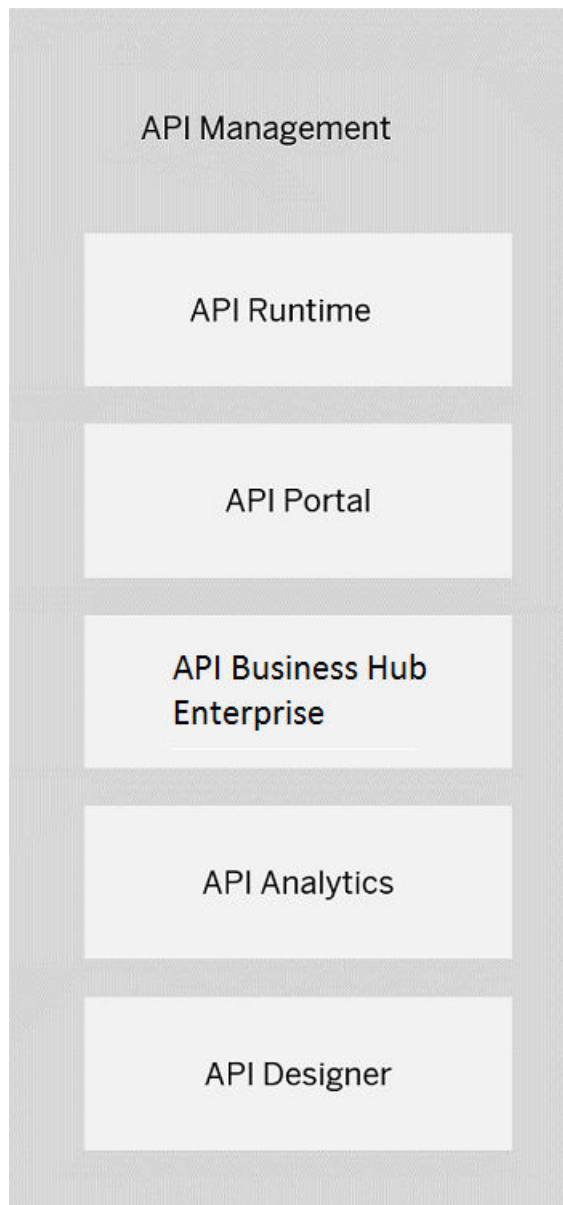
Getting Started

You can provision the API Management capability from the Integration Suite launchpad. For the detailed steps, see [Setting Up API Management Capability from Integration Suite](#)

1.1.1 Components of API Management

API Management component overview

The API Management infrastructure consists of five components: API Runtime, API Portal, API business hub enterprise, API Analytics, and API Designer.



- [#unique_19/unique_19_Connect_42_subsection-im1 \[page 9\]](#)
- [#unique_19/unique_19_Connect_42_subsection-im2 \[page 9\]](#)
- [#unique_19/unique_19_Connect_42_subsection-im3 \[page 9\]](#)
- [#unique_19/unique_19_Connect_42_subsection-im4 \[page 9\]](#)

- [#unique_19/unique_19_Connect_42_subsection-im5 \[page 9\]](#)

Hover over each element for a description. Click the element for more information.

API Runtime

Allows you to deploy and productively use your APIs. Applications consume the API Runtime, request for API authentication, and access.

API Portal

The one-stop-shop to create, secure, and publish API Proxies. This is the place for easy discovery of APIs, and you the API Administrator, can manage, meter, secure your APIs, as well as define and publish rate plans. To know more about using the API Portal see, .

API Business Hub Enterprise

Self-service for application developers to discover, browse, and explore APIs, subscribe to rate plans, and build apps. To know more about using the API business hub enterprise, see [Consume API Proxies \[page 661\]](#).

API Analytics

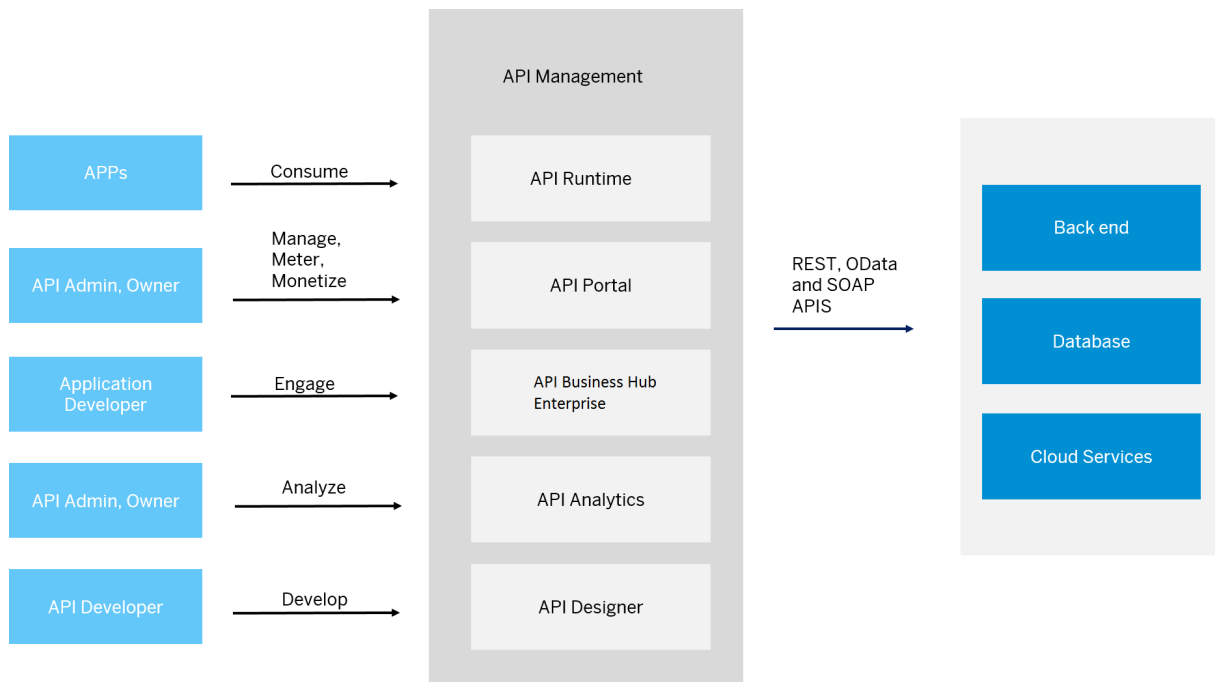
Provides powerful analytical tools to track your API usage. Use API Analytics to collect information on the URL, user ID for API call information, latency data, and so on. To know more about API Analytics, and how you can use it, see [Analyze APIs \[page 701\]](#).

API Designer

API developers can define, implement, and document APIs. It provides open API support, and a variety of outputs can be generated. For more information on the API Designer, see, [here](#).

You can use API Management with one of SAP's numerous in-house API providers as well as any non-SAP APIs. API Management leverages your investment in SAP solutions and can also be integrated with non-SAP solutions. It helps unlock the value of digital assets and enables you to create and deliver content. API Management enables applications to run seamlessly by accessing backend data securely. It provides **one-experience** for managing and monitoring APIs across various data platforms with real-time Analytics.

The following image shows how different stakeholders interact with the various API Management components, and how API Management in turn interacts with the different cloud and on-premise systems.



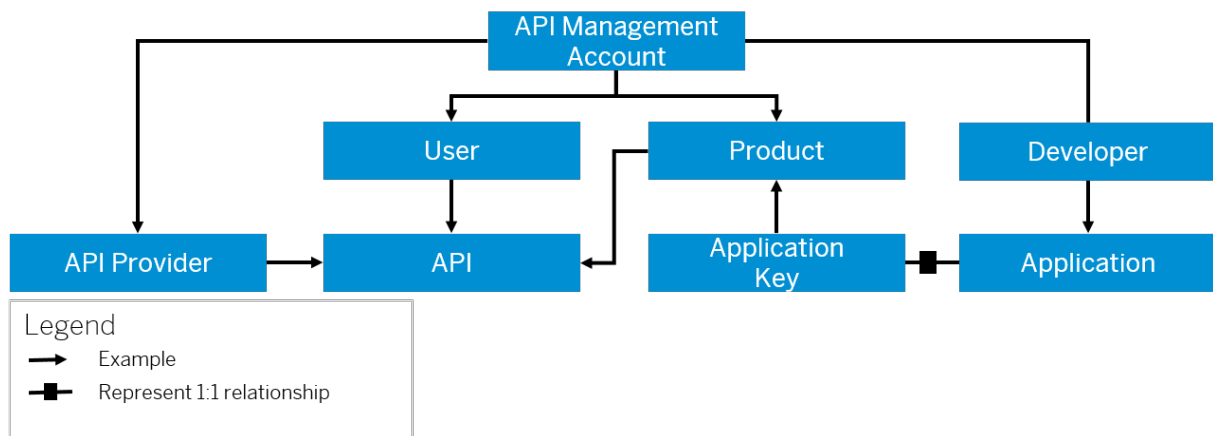
1.1.2 Concepts of API Management

Structure of the API Portal for API Management

Before you start to build APIs, it is important to understand the structure of the API Portal. Its structure defines how APIs, products, applications, users, developers, and accounts are all related to each other within API Management.

The structure of the API Portal comprises:

- API Management Account
- System
- User
- API
- Product
- Developer
- Application
- App Key



The table describes the various entities that comprise the API Portal:



Entity	Definition
API Management Account	An API Management account is the highest level of data hierarchy. An account is a representation of all components including APIs, products, applications, systems, users, and developers.
System	In API Management, System refers to the API provider systems where the actual backend services reside. System could either be an ABAP system, SAP Gateway system, Enterprise Services Repository, or systems that host generic REST services or third party provider systems. API Management allows you to add and manage an API provider system. After you have added a system, you can browse for the APIs in that system.
User	API Management can have multiple users. Different users have different roles and privileges assigned. For example, people who create APIs and products or analyze the metrics or the application consumer who can access the APIs provisioned by API Management.
API	APIs are Application Programming Interfaces. They comprise a set of routines, protocols, and tools for building software applications. APIs define sets of requirements that govern how applications communicate with one another. They facilitate interaction by selectively exposing certain functionalities, allowing different applications, websites, or devices to communicate effectively with each other.
<div style="background-color: #e0e0e0; padding: 5px;"> <p>Note</p> <p>API Management supports OData, REST, and SOAP services.</p> </div>	
Product	A product is a bundle of APIs. It contains metadata specific to your business for monitoring or analytics. For example, all APIs related to CRM can be bundled as one CRM product. API Management collects data for analyzing the products.

Entity	Definition
Developer	<p>One or more developers can create applications in the API Management account. A developer can consume the APIs, but cannot create APIs.</p> <p>To create an application, the developer must have registered the account. After having created an application, the developer uses the app (application) key to consume the APIs.</p>
Application	<p>Applications include the Web or mobile applications that consume the exposed APIs. When you create an application, you select the product to include in this application. For each application that you create, API Management generates an app key and secret. Use this key to gain access to multiple products. Developers create one or more applications using the APIs you expose.</p>
App Key	<p>Based on the authorization mechanism you define for your APIs, the application passes an app (application) key together with every request to your APIs. If that key is valid, the request is permitted. API Management supports different types of authentication, such as a simple API key, OAuth, and so on.</p>

1.1.3 API Services

From API Management, a variety of APIs are offered as services in specific use cases and workflows. You can explore them and try it out in the SAP Business Accelerator Hub in the following url: <http://api.sap.com>.

Services	Description
	<p>You can browse through this API package for API admin services with the required resources.</p>
API business hub enterprise	<p>You can browse through this API package for application developer services that are offered.</p>
Metering	<p>You can now browse through this API package to view metering data for APIs, API Products, and applications in API Portal.</p>

Services	Description
Client SDK	<p>A client software development kit (SDK) is available for developers through a non-commercial license on open source sites.</p> <p>In the API Portal, at the top-right corner, choose <i>Navigation Links</i> () and select <i>Client SDK</i> (). On selecting the client SDK, you are navigated to the maven repository, where you can download this package.</p> <p>For more information, see SAP API Management, 1.6.0 Client SDK .</p> <p>You can also download the package from https://int.repositories.cloud.sap/artifactory/deploy-releases/com/sap/apimgmt/client/sdk/apim-client-sdk/1.6.0/apim-client-sdk-1.6.0.zip . On navigating to this link, select the latest version and choose <i>View All</i>.</p>

1.1.4 Accessibility Features in API Management

To optimize your experience of API Management, we provide features and settings that help you use the software efficiently.

Note

API Management is based on SAPUI5. For this reason, accessibility features for SAPUI5 also apply. See the accessibility documentation for SAPUI5 on SAP Help Portal at [Accessibility for End Users](#).

For more information on screen reader support and keyboard shortcuts, see [Screen-Reader Support for SAPUI5 Controls](#) and [Keyboard Handling for SAPUI5 Elements](#).

1.1.5 Important Notes

Some important notes and announcements.



→ Tip

This is the documentation for API Management for Cloud Foundry. If you are looking for information about the Neo environment, see [here](#).

→ Tip



The English version of this guide is open for contributions and feedback using GitHub. This allows you to get in contact with responsible authors of SAP Help Portal pages and the development team to

discuss documentation-related issues. To contribute to this guide, or to provide feedback, choose the corresponding option on SAP Help Portal:



- [Feedback > Create issue](#) : Provide feedback about a documentation page. This option opens an issue on GitHub.
- [Feedback > Edit page](#) : Contribute to a documentation page. This option opens a pull request on GitHub.

You need a GitHub account to use these options.

More information:

- [Contribution Guidelines](#)
- [Introduction Video: Open Documentation Initiative](#) 
- [Blog Post: Introducing the Open Documentation Initiative](#) 

→ Tip

Contextual help is available for some screens of the SAP Integration Suite. To activate this help, from the top toolbar, choose  *Help*. A panel with help topics opens alongside your current screen to your right. You will also find green  *Help* icons on the screen. Choose these icons to know more about the associated element.

1.2 What's New for SAP API Management Cloud Foundry

Techni- cal Compo- nent	Envi- ron- ment	Title	De- scrip- tion	Action	Lifecy- cle	Typ e	Mo du- lar				
							Lin e of Bu- si- nes s	Bu s Pro ces s	Pro du ct	Lat est Re- sio n	Ava ila- ble as of
API Man- age- ment	Cloud Foundry	Self- Service for Ad- minis- trators to Up- date Custom Domain Certifi- cate	When config- uring a custom virtual host to support TLS, you can specify a key- store or trust- store by using a refer- ence. A refer- ence is a varia- ble that holds the name of the key- store or trust- store, instead of di- rectly specify- ing the name.	Info only	General Availa- bility	Ne w	Tec hn ol- ogy	Not ap- pli- ca- ble	20 24- 04- 06	20 24- 04- 06	

See:
[Working](#)

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			with References (SAP API Management customers choose: Working with References)							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Latency	Availability
API Management	Cloud Foundry	Flow Variables	When executing API proxies created using the on-prem API provider, the backend response may contain URLs that include the API Provider's internal host and port values. These values should not be revealed to API consumers and are not usable.	Info only	General Availability	Network	Not applicable	2024-06	2024-06	

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			To address this issue, the following flow variables have been added to the on-premise proxy so that the API Provider's internal host and port values can be replaced with the virtual host and port values:							
			<ul style="list-style-type: none"> on-premise.eta on-premise.eta 							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			t.h ost							
			<ul style="list-style-type: none"> on-premise.tape.rge.t.port on-premise.tape.rge.t.bah.se.pat.h 							
			These variables ensure that API consumers are not exposed to the internal details of the API provider.							
			See: Flow Varia-							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			bles (SAP API Man- age- ment custom- ers choose: Flow Varia- bles							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	mTLS Authentication	You can now utilize certificate-based credentials to seamlessly migrate API Management subscriptions from Neo to Cloud Foundry, as well as between same or different Cloud Foundry environments.	Info only	General Availability	New	Technology	Not applicable	2024-04	2024-04

See:
[Clone API Management Content](#)

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			and Clone API Management Content for Cloud Foundry to Cloud Foundry Migration (SAP API Management custom- ers choose: Clone API Management Artifacts and Clone API Management Artifacts During							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			Cloud Foundry to Cloud Foundry Migration							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Latency	Availability
API Management	Cloud Foundry	Client SDK version 1.6.0	The outdated libraries in the Client SDK have been upgraded to the latest version.	Info only	General Availability	Network	Not applicable	2024-03-22	2024-03-22	
			See the Client SDK version 1.6.0 in: API Services (SAP API Management custom-ers choose: API Services)							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Self-Service for Administrators to Update Custom Domain Certificate	The recent updates for the configuration of additional virtual hosts in the Cloud Foundry environment. This can be done with either a default domain or a custom domain. Additionally, existing virtual hosts can now have their alias, key-store, keyalias,	Info only	General Availability	New	Technology	Not applicable	2024-09	2024-03-09

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			and trust-store updated. This eliminates the need for the SAP Operations team to manually upload certificates in order to complete the configuration.							
			See: Configuring Additional Virtual Host in Cloud Foundry Environment (SAP							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			API Management	customers choose: Configuring Additional Virtual Host in Cloud Foundry Environment)						

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Manage Access in API business hub enterprise	As an API business hub enterprise admin, you now have the ability to manage access control checks for users to search, discover, and consume the content.	Info only	General Availability	New	Technical	Not applicable	2024-03-09	2024-03-09

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			Manage Access							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Business Processes	Production	Latest Revision	Availability
API Management	Cloud Foundry	Deprecation of Classic Design	Effective June 2024, the classic design of the API business hub enterprise will be deprecated and will no longer be accessible. The new design of the API business hub enterprise will be set as your default design from March 2024.	Info only	Restricted Availability	New	Technology	Not applicable			2024-03-09	2024-03-09

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			See: Configure the API business hub enterprise [New Design] (SAP API Management customers choose: Configure the API business hub enterprise [New Design])							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Threshold Value for Charts in Advanced API Analytics	In the analytics dashboard, the values on the chart for a particular dimension are shown only up to a certain threshold. Any values beyond this threshold are grouped together and labeled as Others . By default, the threshold value	Info only	General Availability	New	Technology	Not applicable	20-24	20-24

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			for the charts is set to 25.							
			See: Find Your Way around Advanced API Analytics Dashboard (SAP API Management custom-ers choose: Find Your Way around Advanced API Analytics Dashboard)							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Deprecation of Weak Client Certificate Chains in API Management	Weak client certificate chains have been deprecated to enhance the security standards. Consequently, the OpenSSL security level has been increased to level 2, which means that weak certificates or certificate chains will no longer be ac-	Info only	Deprecated	Anonymous	Technology	Not applicable	2024-01-20	2024-01-20

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			<p>Accepted by the platform. For a comprehensive definition of security level 2 and further information, please see https://www.openssl.org/docs/man3.0/man3/SSL_CTX_set_security_level.html#DEFAULT-CALL-BACK-BEHAVIOUR information published</p>							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			on non-SAP site.							
			Please note that this change only affects client certificates. If you do not utilize client certificates or mTLS for authentication with the platform, you will not be affected.							
			For additional details on the command, please							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			see 341820							
			1 - Deprecation of Weak Client Certificate Chains in API Management (sap.corp)							
			published on the SAP site.							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Latency	Availability
API Management	Cloud Foundry	Expiration Period for Credentials	The expiration period for the credentials needed to establish a connection to the centralized API business hub enterprise has been extended to 365 days from 65 days. Please make sure to regenerate the credentials and reestablish the connection within	Info only	General Availability	Network	Not applicable	2020-24-01	2020-01-20	

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			<p>this time-frame. However, any credentials generated prior to February 2024 with a validity of 65 days will remain valid for that specific duration. The 365-day time-frame will apply to all newly generated credentials.</p> <p>See:</p>							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			<ul style="list-style-type: none"> Updating the Connection Request Credentials for a Pending Request Updating the Connection Request Credentials 							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			<p>tial</p> <p>s</p> <p>for</p> <p>an</p> <p>Ap-</p> <p>pro</p> <p>ved</p> <p>Re-</p> <p>que</p> <p>st</p> <p>SAP API Management customers choose:</p> <ul style="list-style-type: none"> Up-dati ng the Co nne ctio n Re- que st Cre den tial s for a Pen din 							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			g Re- que st	<ul style="list-style-type: none"> Up- dati ng the Co nne ctio n Re- que st Cre den tial s for an Ap- pro ved Re- que st 						

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Latency	Availability
API Management	Cloud Foundry	Client SDK version 1.5.2	In Cloud Foundry, authentication using X509 certificates and keys is now supported. Instead of using clientid, clientsecret, and tokenurl, you should now use certificate, privateKey, certUrl, and clientid. Additionally, security vulnerabilities re-	Info only	General Availability	Network	Not applicable	2023-12-13	2023-12-13	

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			ported in the SAP NOTE https://me.sap.com/notes/3411067	👉 have been fixed. For more information, please refer to : See the Client SDK version 1.5.2 in: API Services (SAP API Management custom-ers choose: API Services)						

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Latency	Availability
API Management	Cloud Foundry	Policy Template	If there are any default rules or a post-client flow available within the API proxy, they will also be appended to the policy template.	Info only	General Availability	Channel	Not applicable	2023-12-04	2023-12-04	

See: [Apply a Policy Template \(SAP API Management custom-ers choose: Apply a Policy](#)

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			Template							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Business	Provision	Latest	Availability
API Management	Cloud Foundry	Manage external content in API business hub enterprise	The new Manage External Content option in Enterprise Manager enables you to configure additional content, such as Business Data Graphs (BDGs), for display on the API business hub enterprise. For more information, see Manage External Content (New Design) .	Info only	General Availability	New	Technology	Not applicable	2023-12-04	2023-12-04		

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Deprecation of Classic Design of API Business Hub Enterprise	The classic design of the API Business Hub Enterprise will be deprecated soon. Starting from March 2024, the new design of the API Business Hub Enterprise will become the default design. However, you'll still be able to toggle between the new and old	Info only	Deprecated	Anonymous	Technology	Not applicable	2023-12-04	2023-12-04

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			design until June 2024 using the Set Default Design feature in Site Editor . For more information, see See: Customize the Visual Format of the API Business Hub Enterprise(SAP API Management) custom-ers choose: Customize the Vis-							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			ual Format of the API business hub enterprise)							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Lat	Avail-
								Line	est	able
								of	Re-	of
								Business	Pro-	Pro-
								Pro-	du-	sio-
								ces	ct	n
								able		
API Management	Cloud Foundry	Auto-Registration of developers in API business hubs enterprise	If the AuthGroup API. Application Developer role is already assigned to you by the SAP BTP admin or via the IDP Role Collection mapping, you will get automatically registered as an application developer in API business hubs enterprise when	Info only	General Availability	New	Technical	Not applicable	2023-04	2023-12-04



Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			you logon for the first time.	See: Register on API business hub enterprise [page 664] (SAP API Management custom-ers choose: Register on API business hub enterprise)						

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Latency	Availability
API Management	Cloud Foundry	API Revisions	Unique naming pattern for Drafts in API Revisions to clearly differentiate between the deployed draft and the working draft and the drafts originating from revisions.	Info only	General Availability	New	Technical	Not applicable	20-23-30	20-23-30
			See: Creating API Revisions (SAP API Management customers choose:							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Module	Latency	Availability
			Creating API Revisions)							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Latency	Availability
API Management	Cloud Foundry	API Artifact in Edge Integration Cell	With the introduction of the API artifact and the general availability of the Edge Integration Cell, a few navigation changes have been made to the Integration Suite API Management capability. The APIs and Policy Templates, previously located	Info only	General Availability	Network	Not applicable	20-23-30	20-23-30	20-23-30

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latest Revisions	Availability
			under							
			▶ Design							
			▶ n							
			▶ APIs							
			in the							
			left navigation,							
			have							
			been							
			moved							
			to							
			▶ Configuration							
			▶ APIs							
			Additionally,							
			the							
			term							
			"APIs" is							
			now referred							
			to as							
			"API Proxies."							
			Furthermore,							
			"Products"							
			and							
			"Applications",							
			which							
			were							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latest Revision	Availability
			previously found under	 Design  APIs can now be accessed through a new left navigation item called Engage . To know more, see API Development .						

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Lat	Avail-
								of Business	est Re-	ible as
								Pro	Pro	du
								ces	vi-	sio
								ct	n	of
API Management	Cloud Foundry	API Revisions	When publishing products, it's important to note that resources are available from the deployed API, rather than from the latest revision or draft of the API. Additionally, if a deployed resource is not available in the latest revision, you	Info only	General Availability	New	Technology	Not applicable	2023-09-30	2023-09-30

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latest Revision	Availability
			won't be able to attach that resource to the product.							
			The revision name now includes support for parentheses () as an additional special character in the name. For more information, see .							
			<ul style="list-style-type: none"> : API Revisions API Management 							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			em ent sta nda lon e ser vic e: API Re-visions							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Certificate-based Credentials	The connection between the centralised API business hub enterprise and the API portal has been secured with certificate-based authentication. For more information, see Create a Connection Request for the Centralized API business hub enterprise	Info only	General Availability	Network	Not applicable	2023-09-30	2023-09-30	

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			[Classic Design] [page 178] and Create a Connection Request for the Centralized API business hub enterprise [New Design] [page 187] .							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Latency	Availability
API Management	Cloud Foundry	Client SDK	In Cloud Foundry, Authentication using the X509 certificate and key is now supported. Instead of clientid, clientsecret and tokenurl, use certificate, private-key, certUrl and clientid. For more information, refer to the Client SDK version 1.5.0 in	Info only	General Availability	Network	Not applicable	2023-09-13	2023-09-13	

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			API Services							
			[page 12].							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	API Revisions	You can now create a new revision of your API and make necessary changes to the API definitions, policies, and resources without causing any disruption to the already published API.	Info only	General Availability	Network	Not applicable	2023-08-28	2023-08-28	

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latest Revision	Availability
			ous state. For more information, see API Revisions [page 575].							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Latency	Availability
API Management	Cloud Foundry	Graph	Graph is a new and innovative capability of API Management within SAP Integration Suite.It enables you to expose all your business data in the form of a semantically connected data graph, accessed via a single unified and	Info only	General Availability	New	Technical	Not applicable	2023-07-17	2023-07-17

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			powerful API. For more information, see Activating and Managing Capabilities .							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Business Processes	Latency	Availability
API Management	Cloud Foundry	API business hub enterprise: Add Links and Email Addresses Using Markdown	You can use markdown to add links and email addresses to the title and subtitle fields as part of the banner description in Site Editor . For more information, see Customize the Visual Format of the API business hub enterprise [page 677].	Info only	General Availability	Network	Not applicable	2023-07-17	2023-07-17		

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Latest Version of the Tenant Cloning Tool	The 1.7.2 version of the Tenant Cloning Tool is now available. For more information, see Clone API Management Content [page 783] and Clone API Management Content for Cloud Foundry to Cloud Foundry Migration [page 820] .	Info only	General Availability	New	Technology	Not applicable	2023-06-23	2023-06-23

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Business	Provision	Latest	Availability
API Management	Cloud Foundry	Latest Version of the Tenant Cloning Tool	The 1.7.1 version of the Tenant Cloning Tool is now available. For more information, see Clone API Management Content [page 783] and Clone API Management Content for Cloud Foundry to Cloud Foundry Migration [page 820] .	Info only	General Availability	New	Technology	Not applicable	2023-05-22	2023-05-22		

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Lat	Avail-
								of Business	est Re-	ible as
								Pro	Pro	du
								ct	n	of
API Management	Cloud Foundry	Secure options to consume APIs	We have introduced secure options to consume APIs using the API Access Plan for API portal, API business hub enterprise and on-premise connectivity. You can now create service keys with credential types "binding-secret", "instance-secret"	Info only	General Availability	New	Technical	Not applicable	2023-08	2023-05-08

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			and "x509".							
			For more information, see Accessing API Management APIs Programmatically [page 127].							
			Accessing API business hub enterprise APIs Programmatically [page 134]							
			and Accessing On-Premise Systems							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			through API Management [page 150].							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Lat	Avail-
								Line	est	able
								of	Re-	of
								Business	visio-	as
								Products	n	
API Management	Cloud Foundry	Short text field in API products.	In the API portal, a new <i>Short Text</i> field is now available for products. You can use this field to add an introductory text to your products. When you add the introductory text in this field and publish the product, this short text appears in	Info only	General Availability	New	Technology	Not applicable	2023-08	2023-05-08

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			the product tile on the corresponding API business hub enterprise page and is also available on the API proxy details page. For more information, see Create a Product [page 653] .							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business	Modular	Latency	Availability
API Management	Cloud Foundry	API business hub enterprise [New Design] and [Classic Design]	If you have defined the external documentation information for an API in the OpenAPI specification, you can access this information under the Documentation section in the API details page under the Overview tab.	Info only	General Availability	New	Technical	Not applicable	2023-10	2023-04-10

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	API business enterprise	[New Design] The Manage options (for example Manage Users, Manage Do-main] Categories, and so on) on the API business enterprise [New Design] header are now listed under [New DesignEnterprise Manager . Based on the roles assigned to the users,	Info only	General Availability	Technology	Not applicable	2023-02-20	2023-02-20

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			these options will appear under the Enterprise Manager tab. For more information, see Manage Domain Categories [New Design] [page 679].							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	API business hub enterprise [New Design]	You can now configure notifications for providing information to the API business hub enterprise end users on any website updates, events or news items from the API business hub enterprise new user interface. For more information, see Manage Notifi-	Info only	General Availability	New	Technology	Not applicable	2023-02-20	2023-02-20

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latest Revision	Availability
			cations [New Design] [page 681]							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	API business hub enterprise	The mandatory Client ID and Client Secret fields on Edit Credentials popup have been replaced with *API Portal Access Credentials field. For more information, see Updating the Connection Request Credentials for a Pending Request [New Design]	Info only	General Availability	New	Technical	Not applicable	20-23-17	20-23-17

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latest Revision	Availability
			[page 190] and Updating the Connection Request Credentials for a Submitted Request [Classic Design]							
			[page 182] .							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modules	Latency	Availability
API Management	Cloud Foundry	Migrating API Management Subscription from One Cloud Foundry to Another Cloud Foundry Environment	You can choose to migrate an existing API Management Subscription from One Cloud Foundry to Another Cloud Foundry Environment to another API Management Subscription within the Cloud Foundry environment. This can be done to a different	Info only	General Availability	New	Technology	Not applicable	API Management	2023-04-01 to 2023-04-04

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			tenant within the same data center or to a tenant in a different data center. For more information, see Migration of API Management Content between Cloud Foundry Environments [page 817].							

1.2.1 Archive - Release Notes for SAP API Management

Archive of SAP API Management release notes.

[2022 Archives](#) [page 87]

[2021 Archives \[page 105\]](#)

[2020 Archives \[page 112\]](#)

1.2.1.1 2022 Archives

2022

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Line of Business	Business Processes	Module	Product	Version	Latest Release	Availability
API Management	Cloud Foundry	API Business Hub Enterprise [New Design]	If you have added API Business Hub Enterprise as a capability with Integration suite, or if you've subscribed to API Business Hub Enterprise as part of stand-alone API Management subscription, we	Info only	Deprecated	Network	Service	Not applicable	API Management	2022-12-12	2022-12-10		

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latest Revision	Availability
			now have a new design of the user interface for you to experience. For more information, see Consume API Proxies [page 661] and Configure the API business hub enterprise [New Design] [page 675] .							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Decommissioning Concurrent Rate Limit Policy	Support for Concurrent Rate Limit policy has been completely decommissioned. You can no longer create or update an API proxy with Concurrent Rate Limit policy. For more information, see Concurrent Rate Limit [page 255] .	Info only	Deprecated	Network	Service	Not applicable	20-22-24	20-22-24

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latest Revision	Availability
API Management	Cloud Foundry	Consume APIs Using SAP Business Application Studio	The service center in SAP Business Application Studio provides a central entry point to explore products and services from API business hub enterprise. For more information, see Consume API Proxies Using SAP Business	Info only	General Availability	New	Service	Not applicable	2022-07-01	

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latest Revision	Availability
			Application Studio							
			[page 699]							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Kyma	Consuming API Management Service Instance from Kyma	Kyma environment provides a fully managed Kubernetes runtime based on the open-source project "Kyma". You can use the Kyma environment to search and discover API Management, API Portal and API business hub enterprise applica-	Info only	General Availability	Network	Not applicable	Not applicable	2022-07-01	

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			tions. For more information, see Consume API Management Service Instance from Kyma [page 156].							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Decommissioning of Concurrent Rate Limit	The Concurrent Rate Limit policy is being decommissioned. The support for the Concurrent Rate Limit policy will come to an end very soon. If you're still using the policy and wondering which policy to use to best meet your rate-	Info only	Deleted	Anonymous	Service	Not applicable		2022-07-01

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			limiting needs, see Replace Concurrent Rate Limit Policy with Alternative Policies [page 256].							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Optimization of the timeout propagation for API proxies	Previously, the connection to the backend was released only after the default timeout. Now, the connection to the backend is released earlier if the "io.timeout.millis" value is set lower than the default value, thereby improving the performance of the	Info only	General Availability	New	Service	Not applicable	20	22-07-01

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			API proxy. It's recommended that you redeploy the API proxies created based on the on-premise backend system before July 2022. For more information, see Target End-point Properties [page 473].							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Line of Business	Business Processes	Module	Latency	Availability
API Management	Cloud Foundry	Download Open API specification	You can now download the open API specification for the APIs that are part of the API business hub enterprise in JSON format. For more information, see Consume API Proxies [page 661].	Info only	General Availability	Network	Service	Not applicable		20-22-30	

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Service Callout policy	You can now use on-premise type of API Provider in the Service Callout policy. For more information, see Service Callout [page 345] .	Info only	General Availability	Network	Service	Not applicable	20	22-04-02

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Custom Domain Configuration	The custom domain feature is now enabled for API Management as a capability within the Integration Suite product. For more information, see Custom Domain Configuration for API Portal or API business hub enterprise Subscription [page 162].	Info only	General Availability	New	Service	Not applicable		2022-03-11

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latest Revision	Availability
API Management	Cloud Foundry	Skip- ping the cloning of Appli- cation Key and Secret in side by side migra- tion	If you want to skip the cloning of Appli- cation Key and Secret in side in side by side migra- tion, then set the "skip Appli- cations Secret Cloning" flag to true. For more in- forma- tion, see the Pro- cedure section in Clone API Man- age- ment Content [page 783] .	Info only	General Availa- bility	Ne- w	Service process able	Not ap- pli- ca- ble	20- 22- 02- 15	

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			Also, a new version of the tenant cloning tool 1.6.2 is now available. For more information, see Clone API Management Content [page 783].							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Line of Business	Business Processes	Modular Business Processes	Latency	Availability
API Management	Cloud Foundry	Migrating API Management Subscription Created Using Starter Plan Service Instance to a Different Subaccount	If you have In-tergra-tion Suite Pre-mium edition license available in a different sub-account, then API Management design time subscription can be migrated to this different sub-account as well.	Info only	General Availability	Network	Service	Not applicable			2022-01-15

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Modular Business Processes	Latency	Availability
			age-ment Sub-scription Created Using the Starter Plan Service Instance to Different Subaccounts [page 815]. Also, a new parameter has been added to the apim-tct-input.json file. For more information, see Clone API Man-							

Technical Component	Environment	Title	Description	Action	Lifecycle	Type	Business Processes	Module	Availability
			age-ment Content [page 783].						

Parent topic: [Archive - Release Notes for SAP API Management \[page 85\]](#)

Related Information

[2021 Archives \[page 105\]](#)

[2020 Archives \[page 112\]](#)

1.2.1.2 2021 Archives

2021

Technical Component	Capability	Environment	Title	Description	Action	Type	Availability
API Management	Integration Suite	Cloud Foundry	Monitor the Health of Custom Domain Virtual Host Certificates	You can use SAP Cloud Application Lifecycle Management (ALM) application for monitoring the health of API Management certificates. For more information, see Monitor the Health of Custom Domain Virtual Host Certificates Using SAP Cloud ALM [page 780] .	Info only	New	2021-12-12

Technical Component	Capability	Environment	Title	Description	Action	Type	Availability as of
API Management	Integration Suite	Cloud Foundry	Update to the Client SDK	You can now apply policy template to an existing API proxy. For more information, refer to the Client SDK version 1.4.0 in API Services [page 12] .	Info only	New	2021-09-29
API Management	Integration Suite	Cloud Foundry	Applications table on My Workspace	The <i>My Workspace</i> section in API Business Hub Enterprise has been revamped to provide better performance. The Cost Incurred column has been removed from the Applications table. You can view the cost incurred details in the <i>Cost</i> section. For more information, see Create an Application [Classic Design] [page 686] .	Info only	New	2021-09-29
API Management	Integration Suite	Cloud Foundry	SAP Analytics Cloud for API Management	You can use the API Management Reporting Dashboard on SAP Analytics Cloud to monitor API usage and performance through various API metrics and KPIs. For more information see, SAP Analytics Cloud for [page 717] .	Info only	New	2021-08-31
API Management	Integration Suite	Cloud Foundry	Overriding the default update operation for API Proxy of Type ODATA	When discovering an API from the on-premise SAP Gateway system via OData API Provider, for the entities that are defined as "sap:updatable" in the backend service, you can choose the update operation ("PUT" and "PATCH") for OData V2 and OData V4 respectively. For more information, see Overriding the Default Update Operation for API Proxy of Type OData [page 601] .	Info only	New	2021-08-31
API Management	Integration Suite	Cloud Foundry	Product Transport	When transport is triggered for a Product, all the entities of the Product get transported along with the Product. For more information, see Transporting a Product from Source to Destination [page 646] .	Recommended	New	2021-08-09
API Management	Integration Suite	Cloud Foundry	API Designer	For a given Resource, you can use the API Designer to detect and correct the errors in swagger definition, and save the changes. For more information, see the note in step 14 in Create an API Proxy [page 478] .	Info only	New	2021-08-09

Technical Component	Capability	Environment	Title	Description	Action	Type	Availability as of
API Management	Integration Suite	Cloud Foundry	Save and Deploy API Proxy	<p>Saving is a design time activity; at this stage, multiple aspects of the proxy might change. Until all the changes made to the proxy are considered and are finally saved, the proxy should not be deployed.</p> <p>Action: Previously, editing and then saving the changes in an already deployed API, would deploy the changes in runtime. Now, after saving the changes you've made to the API proxy, you have to choose <i>Deploy</i> for the latest changes to reflect in runtime. For more information, see Edit an API Proxy [page 585].</p> <p>Notes:</p> <ul style="list-style-type: none"> API proxies always get imported to the destination API portal in the deployed state. For more information, see Transporting an API Proxy from Source to Destination [page 640]. Additionally, APIs attached to the Product get imported to the destination API portal in the deployed state. For more information, see Transporting a Product from Source to Destination [page 646]. When an API proxy is transported or exported individually or as a part of a Product, by default, it gets imported to the target in the deployed state. For more information, see Import an API Definition [page 534]. If you try to publish a Product that has an API with saved changes attached to it, you get a warning message that there are changes in the APIs that aren't deployed yet. Similarly, you'll receive a warning message if you try to publish a Product which has multiple APIs attached to it, and few of these APIs have changes that are saved but not deployed. For more information, refer the note in step 9 in Create a Product [page 653]. 	Required	New	2021-08-09

Technical Component	Capability	Environment	Title	Description	Action	Type	Availability as of
API Management	Integration Suite	Cloud Foundry	Auditing and Logging Information	Here you can find a list of the security events that are logged by TECHNICAL COMPONENT. For more information, see Auditing and Logging Information for API Management [page 737] .	Info only	New	2021-08-09
API Management	Integration Suite		API Business Hub Enterprise	<p>You can now update the credentials used to establish a connection between the API portal and the API Business Hub Enterprise for a submitted request and an approved request.</p> <p>Action: To update the credentials, see Updating the Connection Request Credentials for a Submitted Request [Classic Design] [page 182] and Updating the Connection Request Credentials for an Approved Request [Classic Design] [page 185].</p>	Recommended	New	2021-07-09
API Management	Integration Suite	Cloud Foundry	Migrating API Management Subscription Created Using the Starter Plan Service Instance	<p>You can now to migrate the design-time components that you have in the Neo environment, which was previously set up using Starter Plan instance, to the Cloud Foundry environment, keeping the runtime components as is.</p> <p>Action: To migrate the design-time components from Neo to Cloud Foundry, see Migrating API Management Subscription Created Using the Starter Plan Service Instance [page 813]</p>	Recommended	New	2021-07-06
API Management	Integration Suite	Cloud Foundry	Custom Domain Configuration for API Portal or API Business Hub Enterprise Subscription	To complete the process of configuring a custom domain for the API Portal or the API Business Hub Enterprise application using the Custom Domain Service in the SAP BTP Cloud Foundry environment, you need to contact the SAP API Management operations team. For more information, see Custom Domain Configuration for API Portal or API business hub enterprise Subscription [page 162] .	Recommended	New	2021-06-25

Technical Component	Capability	Environment	Title	Description	Action	Type	Availability
API Management	Integration Suite	Cloud Foundry	Update to the Client SDK	You can now export policy templates from the API portal. Action: For more information on how to export the policy templates, refer to the Client SDK version 1.3.0 in API Services [page 12] .	Recommended	New	2021-06-14
API Management	Integration Suite	Cloud Foundry	Generate client ID from API Portal UI	You can now generate the credentials from the API Portal to establish the connection with centralized API Business Hub Enterprise. Action: See the Next Steps section in Set Up API Portal Application [page 120] . See the Prerequisites section and the Note in the Results section in Create a Connection Request for the Centralized API business hub enterprise [Classic Design] [page 178] .	Recommended	New	2021-06-14
API Management	Integration Suite	Cloud Foundry	Configuring Load Balancing for API Proxy from API Portal.	You can now configure the load balancer to distribute the load efficiently across multiple API providers. For more information, see Configuring Load Balancing [page 619] .	Recommended	New	2021-06-14
API Management	Integration Suite	Cloud Foundry	Region specific NAT IP addresses	To get region specific NAT IP addresses (egress, IPs for request from API Management), raise a support ticket. For more information, see Region-Specific IP Addresses Available for API Management Cloud Foundry Environment [page 163] .	Info only	Change	2021-06-14
API Management	Integration Suite	Cloud Foundry	Converting Externally Managed APIs to Internally Managed APIs	You can convert an external API, whose lifecycle isn't be managed by SAP API Management to an internal API so that management capabilities can be enabled for that API. Action: To convert an external API to an internal API, see Converting Externally Managed APIs to Internally Managed APIs [page 589] .	Recommended	New	2021-05-13

Technical Component	Capability	Environment	Title	Description	Action	Type	Availability as of
API Management	Integration Suite	Cloud Foundry	Updating the Connection Request Credentials	There can be instances where you have to update the credentials that you've used to establish a connection between the API portal and the API Business Hub Enterprise. For more information, see Updating the Connection Request Credentials for a Submitted Request [Classic Design] [page 182] .	Recommended	New	2021-04-12
API Management	Integration Suite	Cloud Foundry	Perform Additional Tasks in API Designer	To download the API swagger specifications, choose Download and select JSON or YAML format. For more information, see Perform Additional Tasks in API Designer [page 532] .	Info only	New	2021-04-12
API Management	Integration Suite	Cloud Foundry	Transporting API Providers, Certificates, and Key Value Maps	You can now trigger the transport of API Provider, Key Store Certificate, Trust Store, and Key Value Maps individually. Action: To trigger the transport of API Provider, Key Store Certificate, Trust Store, and Key Value Maps individually, see Triggering Content Transport Using SAP Cloud Transport Management Service [page 640] .	Recommended	New	2021-04-12
API Management	Integration Suite	Cloud Foundry	Update to the Client SDK	Two new commands to create API Proxies either by providing parameter information or by uploading JSON files have been added. For more information, see API Services [page 12] .	Info only	New	2021-03-16
API Management	Integration Suite	Cloud Foundry	Content Transport Using Cloud Transport Management Service	You can now use the SAP Cloud Transport Management service for exporting, importing, and shipping the API Management content from the Development or Test environment to Production environment. Action: To transport API Management content from the Development or Test environment to Production environment, see Transport APIs and Its Related Artifacts [page 620] .	Recommended	New	2021-03-16

Technical Component	Capability	Environment	Title	Description	Action	Type	Availability
API Management	Integration Suite	Cloud Foundry	Configuring HealthMonitor and Load Balancer for API Proxy using .Zip	Configure health monitor and the load balancer to put the active target server in rotation and to distribute the load efficiently across multiple servers. For more information, see Load Balancing Across API Providers [page 615] .	Recommended	New	2021-03-16
API Management	Integration Suite	Cloud Foundry	UI changes in the SAP BTP Cockpit	A new path has been added on the UI to subscribe to the application plan for API portal and API Business Hub Enterprise. For more information, refer to the step 3 in the following topics: Set Up API Portal Application [page 120] Set Up API business hub enterprise Application Using the Standalone Tile [page 125] .	Info only	Change	2021-02-15
API Management	Integration Suite	Cloud Foundry	API Business Hub Enterprise	You can now maintain a centralized API catalog in one API Business Hub Enterprise that accepts contents like API proxies, API products, and so on, from multiple API portals. Action: You can select multiple products published from the same portal but you can't select products published from different portals. For more information, see Centralized API business hub enterprise [Classic Design] [page 177] and Create an Application [Classic Design] [page 686] .	Recommended	New	2021-02-15
API Management	Integration Suite	Cloud Foundry	Discover REST and SOAP APIs	Discover REST and SOAP APIs along with OData APIs while creating a proxy for Integration flow. For more information, see Creating an API Proxy using SAP Cloud Integration API Provider [page 603] .	Info only	New	2021-01-18

Parent topic: [Archive - Release Notes for SAP API Management \[page 85\]](#)

Related Information

[2022 Archives \[page 87\]](#)

[2020 Archives \[page 112\]](#)

1.2.1.3 2020 Archives

Technical Component	Capability	Environment	Title	Description	Type	Available as of
API Management	Integration Suite	Cloud Foundry	Edit APIs with an in-built API designer	You can edit your APIs using the API designer, which is now embedded in the API Portal. For more information, see Edit an API Proxy [page 585] .	New	2020-12-23
API Management	Integration Suite	Cloud Foundry	List externally managed APIs on the API-Portal	You can now import and list externally managed APIs on the API Portal. For more information, see Externally Managed APIs [page 588]	New	2020-12-02
API Management	Integration Suite	Cloud Foundry	Consume Integration flows more securely with OAuth Client credentials support for CPI Providers.	You can now use OAuth2ClientCredentials when creating an API provider. For more information see, Create an API Provider [page 538]	New	2020-12-02
API Management	Integration Suite	Cloud Foundry	API state can be entered during import, and is available during export of an API.	For more information on the details of the API state to be provided during import, see Import an API Definition [page 534] .	New	2020-12-02

Technical Component	Capability	Environment	Title	Description	Type	Available as of
API Management	Integration Suite	Cloud Foundry	API Runtime has been updated.	<p>The update in the API Runtime has caused the following changes:</p> <ul style="list-style-type: none"> In the JWT policy, validation may fail if the RSA keys are smaller than 2048 bits. The Concurrent Rate Limit Policy has been deprecated. In the ExtractVariables policy when an XML variable is not resolved via an XPath expression, an error occurs. So, continueOnError should be set to true or IgnoreUnresolvedVariables set to true to allow execution of the policy. 	Changed	2020-10-27
API Management	Integration Suite	Cloud Foundry	Client SDK is now available.	The client SDK is now available, for more information see API Services [page 12] .	New	2020-10-27
API Management	Integration Suite	Cloud Foundry	Advanced API Analytics	Advanced API Analytics brings to you the all new analytics dashboard, providing handy and powerful analytical reporting tools to track your API performance and usage. For more information, see Advanced API Analytics [page 706] .	New	2020-10-27
API Management	Integration Suite	Cloud Foundry	Shadow user creation	There is a new process for shadow user creation, for more information, see Shadow Users [page 176]	Changed	2020-10-21
API Management	Integration Suite	Cloud Foundry	Versioning	You can now version your APIs. For more information, see API Versioning [page 573]	New	2020-10-08
API Management	Integration Suite	Cloud Foundry	Migration from Neo environment to Cloud Foundry	You can now choose to clone the API Portal and API business hub enterprise entities at different times during migration. For more information, see Clone API Management Content [page 783]	New	2020-10-08
API Management	Integration Suite	Cloud Foundry	Embedded API Designer	The API Designer is now embedded within the API Portal, allowing you to create and update your APIs in the same space. You will find some changes in the API designer, such as a shift in the editor to the right side of the screen, and a change in the theme, moving to a brighter background to align with the API portal.	Changed	2020-10-08

Technical Component	Capability	Environment	Title	Description	Type	Available as of
API Management	Integration Suite	Cloud Foundry	Create an API for an Integration Flow	Users can discover Integration Flows through the Cloud Integration API Provider and generate APIs for the same. For more information, see Creating an API Proxy using SAP Cloud Integration API Provider [page 603] .	New	2020-09-09
API Management	Integration Suite	Cloud Foundry	Create an API Provider of type Cloud Integration	Users can now create an API Provider of type "Cloud Integration" to connect to a Cloud Integration system, discover Integration Flows through the API Provider and generate APIs for the same. For more information on creating an API Provider of type Cloud Integration, see Create an API Provider [page 538] .	New	2020-09-09
API Management	Integration Suite	Cloud Foundry	API Services	From API Management, a variety of APIs are offered as services in specific use cases and workflows. For more information, see API Services [page 12]	New	2020-08-11
API Management	Integration Suite	Cloud Foundry	Request for an Additional Virtual Host in Cloud Foundry Environment	Create a new virtual host or update an alias for an existing virtual host in the Cloud Foundry environment. For more information, see Configuring Additional Virtual Host in Cloud Foundry Environment [page 124] .	New	2020-08-11
API Management	Integration Suite	Cloud Foundry	API Proxy States	As an API Management administrator, you can set states for an API proxy while creating or updating the API proxy. For more information, see API Proxy States [page 580] .	New	2020-08-11
API Management	Integration Suite	Cloud Foundry	OpenAPI Specification 3.0 in API Management	API Management now supports OpenAPI Specification (OAS) 3.0. For more information, see OpenAPI Specification 3.0 [page 489] .	New	2020-08-11

Technical Component	Capability	Environment	Title	Description	Type	Available as of
API Management	Integration Suite	Cloud Foundry	Migration Assistance for API Management from Neo to Cloud Foundry Environment	You can now choose to migrate your API Management artifacts from an existing API Management subscription in the Neo environment to another API Management subscription in the public cloud infrastructures (hyperscalers) within the Cloud Foundry environment. For more information, see Migrating API Management from Neo to Cloud Foundry Environment [page 781] .	New	2020-08-06
API Management	Integration Suite	Cloud Foundry	API Access plan for API business hub enterprise	Take a look at the newly introduced API Access plan for the API business hub enterprise in the Cloud Foundry environment. Creating a service instance using this plan enables you to use APIs to interact with the API business hub enterprise. For more information, see Accessing API business hub enterprise APIs Programmatically [page 134] .	New	2020-07-23
API Management	Integration Suite	Cloud Foundry	API Access plan for API Portal	Take a look at the newly introduced API Access plan for the API Portal in the Cloud Foundry environment. Creating a service instance using this plan enables you to use APIs to interact with the API Portal. For more information, see Accessing API Management APIs Programmatically [page 127] .	New	2020-07-23
API Management	Integration Suite	Cloud Foundry	On-premise Connectivity plan	Introduced On-Premise Connectivity plan in the Cloud Foundry environment. Creating a service instance using this plan helps you to obtain a service key to enable principal propagation. For more information, see Accessing On-Premise Systems through API Management [page 150] .	New	2020-06-04
API Management	Integration Suite	Cloud Foundry	API Provider of type <i>On Premise</i>	Create an API Provider of type <i>On Premise</i> to connect to an on-premise system via Cloud Connector. For more information, see Create an API Provider [page 538]	New	2020-06-04

Technical Component	Capability	Environment	Title	Description	Type	Available as of
API Management	Integration Suite	Cloud Foundry	API Management as Route Service plan	Introduced API Management as Route Service plan in the Cloud Foundry environment. Creating a service instance using this plan helps you in managing the Cloud Foundry applications. For more information, see Managing Cloud Foundry Micro-services through API Management [page 147] .	New	2020-06-04
API Management	Integration Suite	Cloud Foundry	API Provider of type Open Connectors	You can now create an API Provider of type Open Connectors to connect to third-party APIs. For more details, see Create an API Provider [page 538]	New	2020-06-04
API Management	Integration Suite	Cloud Foundry	Creating Custom Role	You can now create a custom role for API products in API Management. Take a look at Creating a Custom Role [page 160] to know more. You can also assign permission to a product via UI. For more information, see Assign Permission to a Product via UI [page 658] .	New	2020-03-16
API Management	Integration Suite	Cloud Foundry	Unsubscribing to the API Management service	If necessary, users can unsubscribe to the API portal and API business hub enterprise applications. For more details, see Cancel API Management Service Subscription [page 177] .	New	2020-03-16
API Management	Integration Suite	Cloud Foundry	Setting up API Portal and API business enterprise applications.	You can now set up your API Portal and API business hub enterprise applications on Cloud Foundry environment. Take a look at the initial setup of API Portal application Set Up API Portal Application [page 120] . For setting up the API business hub enterprise application, see Set Up API business hub enterprise Application Using the Standalone Tile [page 125] .	New	2020-02-28

Parent topic: [Archive - Release Notes for SAP API Management \[page 85\]](#)

Related Information

[2022 Archives \[page 87\]](#)

[2021 Archives \[page 105\]](#)

1.3 Patch Releases for API Management

This topic provides information on patch releases for API Management that are provided for bug fixes.

March 2024

Software Increment: 2313

Technical Component	Software Version	Description
API Management	1.165.3	The prefix "custom_" has been added to all custom metrics during creation. Additionally, the Policy Validator for the StatsCollector will check if a custom metric already exists and update it accordingly with the "custom_" prefix in the StatsCollector.

January 2024

Software Increment: 2310

Technical Component	Software Version	Description
API Management	1.162.7	<p>The following issues have been identified:</p> <ul style="list-style-type: none">In the event of an onboarding failure after successfully creating a keystore for opiproxy, the same step was being executed again.When navigating to product details from the product list in the Application details page, particularly when the product has a different name and title in API business hub enterprise the following issues were observed:<ul style="list-style-type: none">Users are unable to navigate to the product details from the product list if the product has a different name and title.The APIProduct uses the name as the primary key in API business hub enterprise, but the Application details page uses the title to navigate, which resulted in an error.The rate limiting for the SAP PKI Certificate Service on all Public & Private Cloud landscapes went live on November 16, 2023, according to the certificate service team. So adjustments need to be made to accommodate the rate limiting. <p>This patch fixes the above issues.</p>

Technical Component	Software Version	Description
API Management	1.162.6	In the Neo platform, the onboarding process for new developers in runtime was not functioning properly. This patch fixes the issue.
API Management	1.162.5	Request initiated for all services to adopt the latest xsuaa security patch.

1.3.1 Patch Archive 2021

October 2021

Software Version: 1.135.*

Technical Component	Software Version	Description	Available as of
API Management	1.135.3	Error messages were popping up on the API Portal Onboarding settings page, and the page was not accessible. Backend configurations were applied to resolve this issue.	2021-10-20

September 2021

Software Version: 1.134.*

Technical Component	Software Version	Description	Available as of
API Management	1.134	<p>The following issues have been fixed with this patch:</p> <ul style="list-style-type: none"> While importing a certificate, the common name of the root certificate was displayed in the portal instead of the common name of the certificate. The data inconsistency encountered by the users while importing certificates has been resolved. When multiple IDPs were configured, users couldn't approve the new developer registration requests in their Production environment, if the user ID of the developers weren't their email IDs. With this patch, such restrictions on the User ID have been removed, and the users can approve the new developer registration requests in their Production environment. 	2021-08-31

August 2021

Software Version: 1.133.*

Technical Component	Software Version	Description	Available as of
API Management	1.133.7	The deployment of an API Proxy would get timed out and fail when large number of Products were associated with it. This patch ensures that in such exceptional cases the deployment of the API Proxy wouldn't fail due to timeout.	2021-08-18
API Management	1.133.6	<p>The following issues have been fixed with this patch:</p> <ul style="list-style-type: none"> Developer registrations in API Business Hub Enterprise would fail when multiple IDPs were associated with the API Management subaccount, and the user ID of the user being onboarded was alphanumeric and not an email ID. The deployment of API Proxies would fail, if the API Proxies in the published Products had access control permissions defined. API Proxy POST/PUT requests with gzip compressed content would fail. Import of API Proxy would fail if it's chained with another API Proxy. To resolve this issue, schema validation for the API Proxy chaining import flow has been enabled. 	2021-08-13

1.4 Configure

Before SAP API Management can be used, the tenant (customer) administrator provisions the API Management applications for the users.

This section includes links to concepts and activities required to set up and start using API Management as a service on Cloud Foundry.

1.4.1 Initial Setup

The two applications that are provisioned are the API portal and API business hub enterprise.

Note

- All activities in this section are performed by the tenant (customer) administrator.
- To subscribe to multitenant application from the Subscriptions page in the SAP BTP cockpit, see [Subscribe to Multitenant Applications in the Cloud Foundry Environment Using the Cockpit](#).
- During the onboarding process, the tenant (customer) should expect API calls from the SAP API Management operations team. These calls may be manual or automated and are made to ensure completeness of the onboarding process.

This section covers the following activities:

- [Set Up API Portal Application \[page 120\]](#)
- [Set Up API business hub enterprise Application Using the Standalone Tile \[page 125\]](#)
- [Account Members \[page 158\]](#)
- [User Roles in API Management](#)

→ Remember

The Cloud Foundry setup is different from the NEO setup. The tutorials apart from the setup remains same.

Related Links:

- [Availability of SAP BTP Regions and Services](#)

1.4.1.1 Set Up API Portal Application

To create APIs, products, import policy templates, and view applications, set up the API portal application.

Context

Depending upon the license you hold, you can set up the *API Management*, *API Portal* capability from the **Integration Suite** launchpad. For more information, see [Enable API Management Capability](#). You can also use

the standalone tile for **API Management, API Portal** to subscribe to the API Portal application. See, [Set Up API business hub enterprise Application Using the Standalone Tile \[page 125\]](#).

Note

- You should either have **Integration Suite** subscription or **API Management, API Portal** tile visibility to set up the API Portal application.
- If you're subscribing to the *API Management, API portal* in the Cloud Foundry subaccount, either by using the **Integration Suite** launchpad or the *API Management, API portal* standalone tile, ensure that you don't have an instance of starter plan created in the same subaccount.
- API Management capabilities from Integration Suite and API Management subscriptions can't coexist in a subaccount.

[Setting Up API Portal Application Using API Management Standalone Tile \[page 121\]](#)

You can provision the API portal using the **API Management, API Portal** standalone tile from the SAP BTP cockpit.

1.4.1.1.1 Setting Up API Portal Application Using API Management Standalone Tile

You can provision the API portal using the **API Management, API Portal** standalone tile from the SAP BTP cockpit.

Prerequisites

Note

If you're a new user, you can't subscribe to the API Management, API portal service independently anymore. To provide a comprehensive integration experience, API Management is only available as a capability of the SAP Integration Suite. For a new subscription of API Management, API portal subscribe to SAP Integration Suite. For more information, see [Activating and Managing Capabilities](#). For visual instructions on how to set up and configure API Management capability from Integration Suite, see [Set Up API Management from Integration Suite](#) tutorial published on SAP site.

- You already have a subaccount and have enable the Cloud Foundry environment in this subaccount. For more information, see [Create a Subaccount](#).
- An *API Management, API portal* entitlement has been created for your subaccount. For more information, see [Configure Entitlements and Quotas for Subaccounts](#).

Context

You should have API Management, API portal subscription to set up the API portal application.

Note

Ensure that you don't have an instance of a starter plan created in the same subaccount where you plan to create an API Management, API portal subscription. Also, note that API Management capabilities from Integration Suite and API Management subscriptions using the stand-alone tile can't coexist in the same subaccount.

Perform the step-by-step instructions to set up the API portal application. However, you can also refer the following video for visual instructions:

Procedure

1. Log on to SAP BTP Cockpit and navigate to your subaccount.
2. Navigate to *Service Marketplace* search for *API Management, API portal* tile and choose *Create*.

Alternatively, navigate to **Services > Instances and Subscriptions**, and choose *Create*.

3. On the *New Instances and Subscriptions* dialog, choose Integration Suite as the *Service*, select the *Standard* plan from the dropdown list, and choose *Create*.

Wait for the subscription to complete successfully.

4. Choose *View Subscription* on the *Creation in Progress* dialog.

Check the status of the submission in subscriptions section on the *Instances and Subscriptions* page. If the subscription is successful you'll notice the status of the *API Management, API portal* changes to *Subscribed*.

5. To access the API Management, API portal, you must first assign the *APIManagement.Selfservice.Administrator* role to yourself.

Note

If you choose *Go to Application* without assigning the *APIManagement.Selfservice.Administrator* role, an application authentication error appears. If the error persists after assigning the role, clear your web browser cache, and log out of the application and log on again.

To assign the role:

1. On the navigation pane, under *Security*, choose *Users*.
2. Select your Username and under *Role Collections* section, choose *Assign Role Collection*.
3. In the resulting dialog box, select the *APIManagement.Selfservice.Administrator* role and choose *Assign Role Collection*.
6. Navigate back to the *Instances and Subscriptions* page, choose *API Management, API portal*, select **...** *Actions* and choose *Go To Application*.

You're navigated to the *API Portal*.

7. On the *Configure the API Management Service* screen, select the following and choose *Set Up*:
 - Select the *Account type*:
 - Select the *Non Production* account type for non-business critical activities, such as integrating test systems, testing new scenarios, performance testing, and sandbox activities.

- Select the *Production* account type for business critical usages, such as integrating production systems, and productive APIs.
- In the *Virtual Host* section, enter the *Host Alias*.
- Provide an email ID in the *Notification Contact* field to receive updates.

In the *Set-up Confirmation* window, review the provided details and choose *Confirm* to start the onboarding process.

You're redirected to a progress window, which states *API Management Service Setup In Progress*.

The *Configuration* process is triggered, where the necessary resources are provisioned for you. It's followed by *Testing the Setup* process, where a simple API Proxy is deployed and invoked to check that everything is set up properly.

When the processes complete, the indicators turn green to indicate that the processes are successful. A *Release Notification* mail is sent out to the email IDs provided in the *Configure the API Management Service* screen. This email contains details of the newly set up API Management service on your account.

The API Management, API portal application is now configured.

8. You must log out of the *API Portal* and login again.

Now, you can create APIs, build API proxies as a service provider, or use APIs and other convenient services.

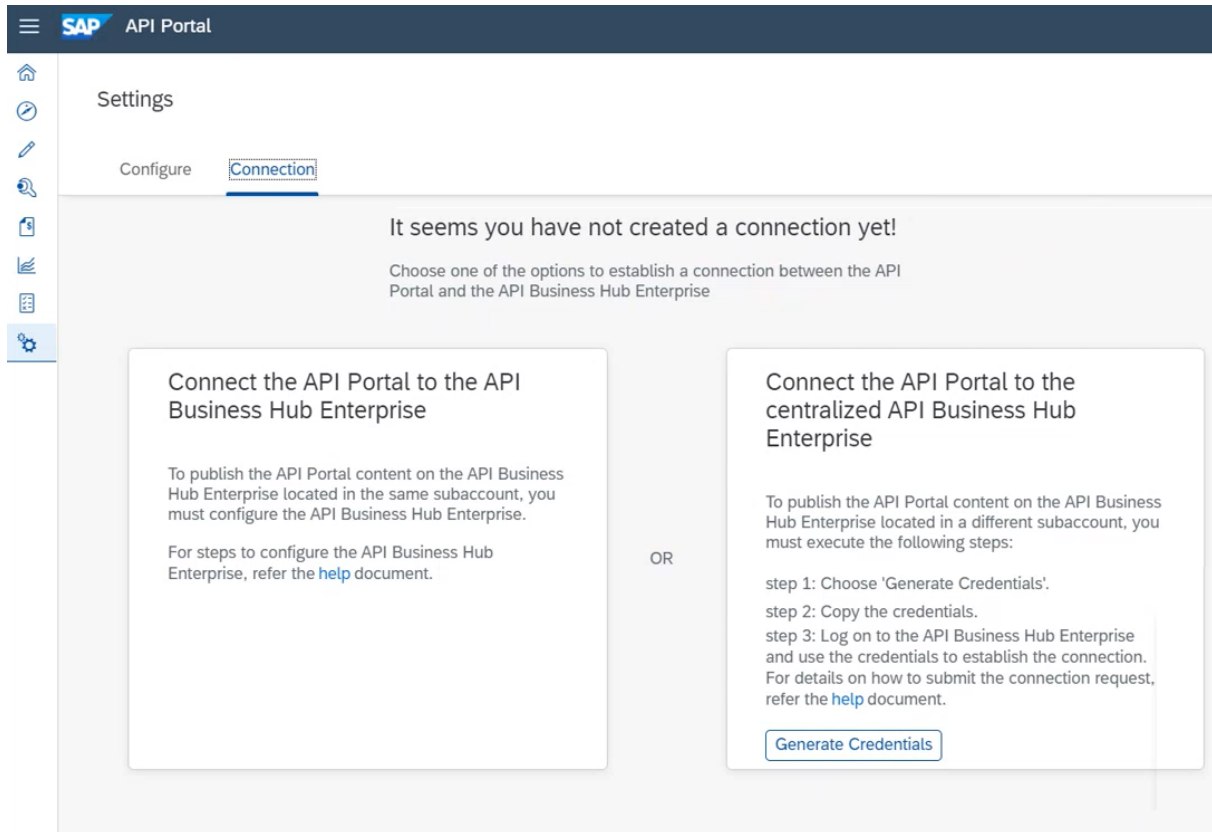
Results

The API portal is now configured. Log on to the API portal again. You can now create APIs, build API proxies as a service provider, or use APIs and other convenient services.

Next Steps

To start publishing the API portal content, you must enable the API Business Hub Enterprise. To publish the API portal content on the API Business Hub Enterprise located in the same subaccount, see [Set Up API business hub enterprise Application Using the Standalone Tile \[page 125\]](#). To publish the API portal content on the centralized API Business Hub Enterprise, follow the on-screen instructions and see [Centralized API business hub enterprise \[Classic Design\] \[page 177\]](#).

Once API Business Hub Enterprise is set up, navigate to  *Onboarding Settings* and choose *Connection*.



Task overview: [Set Up API Portal Application \[page 120\]](#)

Related Information

[Cancel API Management Service Subscription \[page 177\]](#)

[Build API Proxies \[page 195\]](#)

[Create an API Provider \[page 538\]](#)

[Different Methods of Creating an API Proxy \[page 477\]](#)

[Configuring a Custom Domain for a Virtual Host \[page 170\]](#)

[Configuring Mutual TLs for Virtual Host \[page 174\]](#)

1.4.1.1.1.1 Configuring Additional Virtual Host in Cloud Foundry Environment

A virtual host allows you to host multiple domain names on the API Management capability within Integration Suite.

Create a new virtual host with default domain or custom domain and update alias, keystore, keyalias, and truststore for an existing virtual host in the Cloud Foundry environment.

[Configuring a Default Domain for a Virtual Host \[page 166\]](#)

After successful onboarding, API proxies are assigned a default virtual host URL. Currently, this URL uses the domain "ondemand.com," which is the common domain for the Business Technology Platform. It's prefixed with the subdomain consisting of the subaccount name and the data center where the Integration Suite tenant is onboarded. For example, the default host alias could be `https://myaccount...eu10.hana.ondemand.com`.

[Configuring a Custom Domain for a Virtual Host \[page 170\]](#)

The API Management capability enables you to personalize the virtual host URL by configuring a custom domain of your choice. This means that you can have all your APIs displayed as "https://api.bestrun.com/..." if desired. Additionally, you have the option to set up multiple virtual hosts using the same custom domain, such as "https://api1.bestrun.com," "https://api2.bestrun.com," and so on.

[Configuring Mutual TLS for Virtual Host \[page 174\]](#)

You can configure mutual TLS for a virtual host, which validates the identities of both the web server and the web client.

1.4.1.2 Set Up API business hub enterprise Application Using the Standalone Tile

To discover, consume and monitor API from a centralized API catalog, set up the API business hub enterprise application.

Prerequisites

- You already have a subaccount and have enable the Cloud Foundry environment in this subaccount.
- An *API Management, API Business Hub Enterprise* entitlement has been created for your subaccount.
- You have already set up and configured API portal. For instructions, see [Setting Up API Portal Application Using API Management Standalone Tile \[page 121\]](#).

Context

Depending upon the license you hold, you can use the *API Management, API Business Hub Enterprise* stand-alone tile to subscribe to the application, or you can set up the API business hub enterprise capability from the **Integration Suite** launchpad. To set up API business hub enterprise from **Integration Suite**, see .

📘 Note

Ensure that you don't have an instance of a starter plan created in the same subaccount where you plan to create an API business hub enterprise subscription. Also, note that the API Management capabilities from Integration Suite and API Management subscriptions using the stand-alone tile can't coexist in the same subaccount.

Procedure

1. Log on to SAP BTP Cockpit and navigate to your subaccount.
2. Navigate to ► [Services](#) ► [Instances and Subscriptions](#) ▾, and choose [Create](#).
3. On the *New Instances and Subscriptions* dialog, choose [API Management, API Business Hub Enterprise](#) as the *Service*, select the *Plan* from the dropdown list, and choose [Create](#).

Wait for the subscription to complete successfully.

4. To access the API business hub enterprise, you must first assign the [AuthGroup.SelfService.Admin](#) role to yourself.

Note

If you choose [Go to Application](#) without assigning the [AuthGroup.SelfService.Admin](#) role, an application authentication error appears. If the error persists after assigning the role, clear your web browser cache, and log out of the application and log on again.

To assign the role:

1. On the navigation pane, under [Security](#), choose [Users](#).
2. Select your username and under [Role Collections](#) section, choose [Assign Role Collection](#).
3. In the resulting dialog box, select the [AuthGroup.SelfService.Admin](#) role and choose [Assign Role Collection](#).
5. Navigate back to the [Instances and Subscriptions](#) page, choose [API Management, API Business Hub Enterprise](#), select [⋮ Actions](#) , and choose [Go To Application](#).

You're navigated to the [API Business Hub Enterprise](#).

6. Log on to the API business hub enterprise application with your IDP user credentials. To register to the API business hub enterprise as an Application developer, see [Register on API business hub enterprise \[page 664\]](#).

Results

You're registered as an application developer on the API business hub enterprise. You can now view the products available in the catalog store.

Related Information

[Configure the API business hub enterprise \[Classic Design\] \[page 671\]](#)

[Create an Application \[Classic Design\] \[page 686\]](#)

1.4.1.3 API Management Service Plans

The functionality of the API Management capability is typically managed using the Integration Suite (UI) application, which is the primary focus of this user guide. However, you can also call API Management APIs, manage Cloud Foundry applications, and connect to an on-premise backend system by means of service plans.

The API Management services on Cloud Foundry provides different capabilities through the following plans:

[Accessing API Management APIs Programmatically \[page 127\]](#)

The *apiportal-apiaccess* plan offers external applications the ability to access the public APIs of the Integration Suite API Management capability. These APIs are used by the external applications to perform CRUD operations on API Management features like API proxies or products. These APIs are built on REST and OData principles and are extensively documented on the [Business Accelerator Hub](#) .

[Accessing API business hub enterprise APIs Programmatically \[page 134\]](#)

The *devportal-apiaccess* plan allows you to access the API business hub enterprise APIs to programmatically onboard developers, create applications, and more.

[Managing Cloud Foundry Microservices through API Management \[page 147\]](#)

The *apim-as-route-service* plan helps you in managing Cloud Foundry applications by including policies such as rate limit, quota. The service instance you create through this plan allows you to bind to the route service and creates an API Proxy. This API Proxy serves in establishing a secure connection with your Cloud Foundry application and all the calls made to the Cloud Foundry application are routed via *API Management, API portal*.

[Accessing On-Premise Systems through API Management \[page 150\]](#)

The *on-premise-connectivity* plan helps in achieving principal propagation while connecting to an on-premise backend system.

1.4.1.3.1 Accessing API Management APIs Programmatically

The *apiportal-apiaccess* plan offers external applications the ability to access the public APIs of the Integration Suite API Management capability. These APIs are used by the external applications to perform CRUD operations on API Management features like API proxies or products. These APIs are built on REST and OData principles and are extensively documented on the [Business Accelerator Hub](#) .

About the Plan

The *apiportal-apiaccess* plan allows you to programmatically import/export API proxies, create products, key value maps. It is especially useful when integrating API Management with a CI/CD process or when migrating from a Neo to Cloud Foundry environment using the migration tool.

The API Access plan allows you to generate a service key by creating a service instance. By creating a service instance, you can generate a service key that includes the application url, clientId, clientSecret, and tokenUrl is used to generate a bearer token with the help of a REST Console. This Bearer Token, along with the application url and API endpoint are used to trigger the API. Therefore, bearer token acts like a key to access the APIs.

Prerequisites

- You've enabled API Management capability using Integration suite. For more information, refer [Subscribing to Integration Suite](#) and [Activating Capabilities](#).
OR
You have subscribed to the standalone *API Management, API portal* tile in the Cloud Foundry environment. For more information, see [Set Up API Portal Application \[page 120\]](#).
- As a subaccount administrator, you additionally need the role of (org member) and *space developer* in the Cloud Foundry space in which the Integration Suite is provisioned.

To enable API access for API Management, API portal execute the steps in the sections below:

Creating a Service Instance in the API Management, API portal

Create a service instance using API Access plan to generate a service key.

1. In your web browser, open the *SAP BTP Cockpit* - <https://eu-access.cockpit.btp.cloud.sap>.
2. From your *Subaccount*, navigate to *Spaces* in your Cloud Foundry environment and choose **Services** **> Service Marketplace**.
3. Choose **API Management, API portal** **> Instances** **> New Instance**.

Note

If you are unable to view the *API Management, API Portal* tile, please check your entitlements. For more information, see [Managing Entitlements and Quotas Using the Cockpit](#).

4. In the *Create Instance* dialog that opens, choose the *apiportal-apiaccess* plan.
5. In the section *Specify parameters*, paste one of the following JSON codes, to assign a specific role. The following roles are supported for the current scenario:
Assign `APIPortal.Administrator` role to access the API portal APIs and perform operations like create, update, delete on various API portal entities as specified in the [SAP Business Accelerator Hub](#).

```
{
  "role": "APIPortal.Administrator"
}
```

Assign `APIPortal.Guest` role to access the API portal APIs in read-only mode. You can view the API portal entities as specified in the API Business Hub.

```
{
  "role": "APIPortal.Guest"
}
```

Assign `APIManagement.SelfService.Administrator` role to configure additional virtual hosts.

```
{
  "role": "APIManagement.SelfService.Administrator"
}
```

6. Click *Next* until you reach the *Confirm* section
7. In the section *Confirm*, enter a unique *Instance Name* and choose *Finish*.

The creation of service instance is successful.

Now, with the help of the created service instance, generate a service key from the steps given below:

Creating a Service Key

1. Choose the created service instance link from the visible list.
2. In the left-hand pane, navigate to **Service Keys** > **Create Service Key**.
3. In the *Create Service Key* dialog that opens, provide a *Name* and *Description* (optional).
4. In the text box enter one of the following payloads as per your requirement:

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
"instance-secret" (without payload)		Low	For instance-secret, the clientSecret generated is same for all the keys.	For admin role: <pre>{ "url": "https:// apiportal- application- url", "tokenUrl": "https:// token- endpoint-url/ oauth/ token", "clientId": "your- client- id", "clientSecret": "xxxxxxxxxxxxx xxxxxxxxxxxxxxx =" }</pre>

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
"instance-secret" (with payload)	<pre>{ "xsuaa":{ "credential-type":"instance-secret" } }</pre>	Low	For instance-secret, the clientSecret generated is same for all the keys.	For admin role: <pre>{ "url": "https:// apiportal- application- url", "tokenUrl": "https:// token- endpoint-url/ oauth/ token", "clientId": "your- client- id", "clientSecret": "xxxxxxxxxxxx xxxxxxxxxxxxxx =" }</pre>
"binding-secret"	<pre>{ "xsuaa":{ "credential-type":"binding-secret" } }</pre>	Medium	For binding-secret, the clientSecret generated for every key is unique.	For admin role: <pre>{ "url": "https:// apiportal- application- url", "tokenUrl": "https:// token- endpoint-url/ oauth/ token", "clientId": "your- client- id", "clientSecret": "xxxxxxxxxxxx xxxxxxxxxxxxxx xxxxxxxxxxxxxx xxxxxxxxxxxxxx =" }</pre>

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
"x509" (certificate based)	<pre>{ "xsuaa": { "credential-type": "x509" }, "x509": { "key-length": 2048 }, "validity": 65, "validity-type": "DAYS" }</pre>	High	For X509, ensure that the credential rotation is done based on the validity provided in the payload. For example, delete and create a new service key every 65 days.	For admin role: <pre>{ "url": "https://xxxxxxxxxxxxx.cfapps.sap.hana.ondemand.com", "certificate": "xxxxxxxxxxxxx", "certurl": "https://xxxxxx.authentication.cert.sap.hana.ondemand.com", "clientId": "xxxxxxxxxxxxx", "privateKey": "xxxxxxxxxxxxx", "tokenUrl": "https://xxxxxx.authentication.sap.hana.ondemand.com/oauth/token" }</pre>

- Click [Save](#). The client credentials like [url](#), [clientId](#), [clientSecret](#), and [tokenUrl](#) details appear for the given instance name.

The application [url](#) is used to make API calls.

The [clientId](#) and [clientSecret](#) are necessary credentials required to fetch the Bearer Token.

The [tokenUrl](#) is used to fetch the bearer token.

Example:

```
url --cert client.crt --key client.key --location --request POST '<URL from the response>' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--header 'Authorization: No Auth \
--data '{"client_id":"clientId"}' `grant_type=client_credentials`
```

The generated credentials (tokenUrl, clientId and clientSecret) can then be used by an application developer to authenticate the client (or user) via OAuth, and access the APIs exposed by the service. Copy the client credentials in a notepad.

Note

Once your client is setup you can use it to authenticate against the x509 endpoint by providing the client certificate and key.

Create the certificate.cer and certificate.key files based on the public and private keys obtained from the x509 credentials respectively.

For certificate.cer file:

1. Copy the "certificate" value starting from -----BEGIN CERTIFICATE----- all the way to -----END CERTIFICATE-----\n (the certificate value might contain multiple certificates). Make sure you copy all the certificates.
2. Paste it in a text editor. Find and replace all the occurrences of \n by \n.
3. Save the file as certificate.cer.

For certificate.key file:

1. Copy the "certificate" value starting from -----BEGIN RSA PRIVATE KEY-----\n... all the way to-----END RSA PRIVATE KEY-----\n
2. Paste it in a text editor. Find and replace all the occurrences of \n by \n.
3. Save the file as certificate.key.

Open a command prompt/terminal in the folder where you have saved the certificate files and execute the following curl command to get the response in the my-oauth-response.json file in the same folder. From this file, you can fetch the bearer token from the value of "access_token".

```
curl --cert certificate.cer --key certificate.key --location --request
POST '<certurl from the servicekey x509 credentials>/oauth/token?
grant_type=client_credentials' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'client_id=<clientId from the servicekey x509
credentials>' \
--cert certificate.cer \
--key certificate.key \
> my-oauth-response.json
```

Note

You can update an already provisioned service instance of an API access plan by performing the following steps:

Prerequisite: You must have the Cloud Foundry CLI installed.

1. Log in to the Cloud Foundry CLI by running the `cf login` command.
2. Select *Org*.
3. Run the following command to update your service instance. `cf update-service <service-instance-name> -c <empty-json-file>.json.`

Sample Code

```
Sample json: {}
```

Next Steps: Obtaining a Bearer Token

In the REST Console:

1. Paste the copied `tokenUrl`. Append `?grant_type=client_credentials` to the `tokenUrl`.
2. Choose `Basic Auth` as the `Authorization` type.
3. Similarly, paste the `clientId` and `clientSecret` in the place of `Username` and `Password`.
4. Make a POST Call.
5. Obtain the Bearer Token from the output and copy it in a notepad.
 - Now, to trigger an API, in the same REST Console, append the API endpoint (obtained from the API portal APIs that are located in the SAP API Management package of API Business Hub) to the `url`.

Note

Currently, the `apiportal-apiaccess` plan allows you to access only the API Management APIs from the `SAP API Management package`.

- Choose `Bearer Token` as the `Authorization` type and paste the copied Bearer Token in the specified space.
- Include payloads, if needed.

Example

This example summarizes the steps that we have executed so far.

To fetch all the available API proxies in your API portal, you can make a call to the URL that appears in the endpoint `<url>/apiportal/api/1.0/Management.svc/APIProxies`.

To successfully trigger this API, you need to enable the 'API portal API Access Plan'. Once the plan is enabled and suitable roles are assigned, you can generate a service key using the created service instance. The service key should include your URL, client ID, client secret, and token URL. Copy the URL and append ``/apiportal/api/1.0/Management.svc/APIProxies`` to it, as shown here: `<url>/apiportal/api/1.0/Management.svc/APIProxies`

In addition, to establish a connection with this API, you will need to generate a bearer token.

Take the token URL and append the grant type to it: `https://token-endpoint-url/oauth/token?grant_type=client_credentials`

Now, execute a GET operation on this API with the Client ID and Client secret as your Basic auth credentials. The bearer token will be embedded in the response code. Next, append the bearer token to the URL as shown here: `<url>/apiportal/api/1.0/Management.svc/APIProxies/<bearer token>`

By calling this URL, you will be able to list all the API proxies available in your API portal.

Parent topic: [API Management Service Plans \[page 127\]](#)

Related Information

[Accessing API business hub enterprise APIs Programmatically \[page 134\]](#)

1.4.1.3.2 Accessing API business hub enterprise APIs Programmatically

The *devportal-apiaccess* plan allows you to access the API business hub enterprise APIs to programmatically onboard developers, create applications, and more.

About the Plan

The service key, consisting of url (application url), clientId, clientSecret, and tokenUrl is used to generate a Bearer Token with the help of a REST client. This Bearer Token, along with the application url and API endpoint, is used to trigger the APIs.

This topic explains how to enable API access for API business hub enterprise.

Prerequisites

- If you've enabled API Management capability using Integration suite, ensure that you've also enabled API business hub enterprise in Integration suite. For more information, refer [Subscribing to Integration Suite](#) and [Activating Capabilities](#). To access API business hub enterprise from Integration Suite, select API business hub enterprise from the *Navigation Links* on the header.

Note

Please ensure that you can access API business hub enterprise before creating an instance.

- You have the **space developer** role assigned to you.
- You have created a service instance under the *Authorization and Trust Management* tile.
 1. In your web browser, open the *SAP BTP Cockpit* - <https://eu-access.cockpit.btp.cloud.sap>.
 2. From your *Subaccount*, navigate to *Spaces* in your Cloud Foundry environment and choose **Services** **Service Marketplace**.
 3. Choose **Authorization and Trust Management** **Instances** **New Instance**.
 4. In the *Create Instance* dialog that opens, choose the *apiaccess* plan.
 5. Click *Next* until you reach the *Confirm* section.
 6. In the section *Confirm*, enter a unique *Instance Name* and choose *Finish*.
- You have created a service key for the service instance above.
 1. Choose the service instance that you created above.
 2. In the left-hand pane, navigate to **Service Keys** **Create Service Key**.
 3. In the *Create Service Key* dialog that opens, provide a name.

4. Click *Save*.

The client credentials like url, clientId, and clientSecret details appear for the given service key.

- You have created a destination of type **OAuth2Credentials** to the XSUAA APIs by using the credentials you derived from creating the service key.

1. From your *Subaccount*, navigate to **Connectivity > Destinations > New Destination**.
2. Choose the service instance that you created above.
3. In the *Destination Configuration* window, provide the details.

Note

You must enter the details exactly as mentioned below:

```
Name: apingmt-platform-access
Type: HTTP
Description:
URL: https://yourxsuaa.authentication.sap.hana.ondemand.com (Provide the
value of the url field from the service key you created above.)
Proxy Type: Internet
Authentication: OAuth2ClientCredentials
Client ID: apiaccess-client_id (Provide the value of the "clientid" field
from the service key you created above.)
Client Secret: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx (Provide the value of the
"clientsecret" field from the service key you created above.)
Token Service URL: https://yourxsuaa.authentication.sap.hana.ondemand.com
(Provide the value of the url field from the service key you created
above.)
Token Service User:
Token Service Password:
```

- For URL, provide the value of the **url** field from the service key you created above.
- For Client ID, provide the value of the **clientid** field from the service key you created above.
- For Client Secret, provide the value of the **clientsecret** field from the service key you created above.
- For the Token Service URL, provide the value of the **url** field from the service key you created above.

4. Click *Save*.

Creating a Service Instance in the API Management, API business hub enterprise

Create a service instance using API Access plan.

1. In your web browser, open the *SAP BTP Cockpit* - <https://account.hana.ondemand.com/cockpit>.
2. From your *Subaccount*, navigate to *Spaces* in your Cloud Foundry environment and choose **Services > Service Marketplace**.
3. Choose **API Management, API Business Hub Enterprise > Instances > New Instance**.
4. In the *Create Instance* dialog that opens, choose the plan as *devportal-apiaccess*.
5. Click *Next*.
6. In the section *Specify parameters*, provide the details as mentioned below, based on the role you require.

The roles that support API access in the API business hub enterprise are `AuthGroup.API.Admin`, `AuthGroup.Content.Admin`, and `AuthGroup.API.ApplicationDeveloper`. Create a service instance with the `AuthGroup.API.Admin` role to access the API business hub enterprise APIs (applications and attributes, API packages, API proxies and products, app developer and metering), and perform operations like create, update, and delete on various API business hub enterprise entities as specified in the [Business Accelerator Hub](#).

```
{
  "role": "AuthGroup.API.Admin"
}
```

Create a service instance with the `AuthGroup.Content.Admin` role to manage the domain categories in API business hub enterprise and add the related products into relevant categories.

```
{
  "role": "AuthGroup.Content.Admin"
}
```

Create a service instance with the `AuthGroup.API.ApplicationDeveloper` role to access the API business hub enterprise APIs (applications, API packages, and API proxies and products), and perform operations like create, update, and delete on various API business hub enterprise entities as specified in the [Business Accelerator Hub](#).

```
{
  "role": "AuthGroup.API.ApplicationDeveloper"
  "developerId": "developerId"
}
```

Note

What is `developerId`:

Providing an invalid or an empty `developerId` throws an error in the service instance creation process.

To successfully create an application via the API business hub enterprise, you must provide a valid `developerId`. This means that you must have already registered as an application developer to the API Management, API business hub enterprise service or you must have been onboarded by your administrator.

- If you have registered to the API Management, API business hub enterprise application, provide your `developerId`. See the section below to know how to obtain your `developerId`.
- If you have not registered to the API Management, API business hub enterprise application, follow the steps in [Register on API business hub enterprise \[page 664\]](#) and try again.
- If you are not registered to the API Management, API business hub enterprise application, and require your admin to onboard you, contact your admin. See [Onboard an Application Developer \[page 663\]](#).

How to obtain the `developerId`:

- If you are a registered developer in the API business hub enterprise, access the following URL in your browser to obtain your `developerId`:

```
https://devportal-url/api/1.0/user
#Response
[ {"Name": "" ,
```



```
"FirstName":"","  
"LastName":"","  
"LoggedOut":false,  
"Email":""}]
```

The **Name** field in the response is your **developerId**.

- If you are an admin and are obtaining the **developerId** for a developer you have already onboarded, pick the **userId** that you provided during the developer onboarding. To view a list of the registered developers, access the following URL in your browser. The **userId** field in the response is the **developerId**.

```
https://devportal-url/api/1.0/registrations?type=registered  
#Response  
autoReLogin: false  
country: ""  
emailId: ""  
firstName: ""  
lastName: ""  
rolesAccess: [{status: "registered", role: "API_ApplicationDeveloper"}]  
0: {status: "registered", role: "API_ApplicationDeveloper"}  
userId: ""
```

Limitation: Self-service onboarding request is not supported for a developer. So, the POST operation under the *API Business Hub Enterprise - Registering Users* tile in the *API Business Hub* cannot be made by the application developer service key. As an alternative, you can invoke this API using the admin service key.

7. In the section *Confirm*, enter a unique *Instance Name*, and choose *Finish*.

The service instance is successfully created and listed in the *Instances* window.

Create a Service Key

Generate a service key for the service instance that you created above:

1. From the *Instances* window, choose the service instance that you created above.
2. In the left-hand pane, navigate to **Service Keys** **Create Service Key**.
3. In the *Create Service Key* dialog that opens, provide a name.

4. In the text box enter one of the following payloads as per your requirement:

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
"instance-secret" (without payload)		Low	For instance-secret, the clientSecret generated is same for all the keys.	<p>For admin role:</p> <pre data-bbox="1193 510 1375 1182"> { "url": "https:// developer- portal- application- url", "tokenUrl": "https:// token- endpoint-url/ oauth/ token", "clientId": "your-admin- client- id", "clientSecret": "xxxxxxxxxxxxx xxxxxxxxxxxxxx =" } </pre> <p>For developer role:</p> <pre data-bbox="1193 1272 1375 1917"> { "url": "https:// developer- portal- application- url", "tokenUrl": "https:// token- endpoint-url/ oauth/ token", "developerId": "developerID- associated- with-the- current- instance", "clientId": "your-dev- </pre>

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
				<pre> client- id", "clientSecret": "xxxxxxxxxxxxx xxxxxxxxxxxxx =" } </pre>

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
"instance-secret" (with payload)	<pre>{ "xsuaa":{ "credential-type":"instance-secret" } }</pre>	Low	For instance-secret, the clientSecret generated is same for all the keys.	<p>For admin role:</p> <pre>{ "url": "https:// developer- portal- application- url", "tokenUrl": "https:// token- endpoint-url/ oauth/ token", "clientId": "your-admin- client- id", "clientSecret": "xxxxxxxxxxxx xxxxxxxx=" }</pre> <p>For developer role:</p> <pre>{ "url": "https:// developer- portal- application- url", "tokenUrl": "https:// token- endpoint-url/ oauth/ token", "developerId": "developerID- associated- with-the- current- instance", "clientId": "your-dev- client- id", "clientSecret": </pre>

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
				<pre> "xxxxxxxxxxxxx xxxxxxxxxx=" } </pre>

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
				<pre> "clientSecret": "xxxxxxxxxxxxx xxxxxxxxxxxxx xxxxxxxxxxxxx xxxxxxxxxxxxx =" } </pre>

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
"x509" (certificate based)	<pre>{ "xsuaa": { "credential-type": "x509" }, "x509": { "key-length": 2048 }, "validity": 65, "validity-type": "DAYS" }</pre>	High	For X509, ensure that the credential rotation is done based on the validity provided in the payload. For example, delete and create a new service key every 65 days.	<p>For admin role:</p> <pre>{ "url": "https://xxxxxxxxxxxxx.cfapps.sap.hana.ondemand.com", "certificate": "xxxxxxxxxxxxx", "certurl": "https://xxxxxx.authentication.cert.sap.hana.ondemand.com", "clientId": "xxxxxxxxxxxxx", "privateKey": "xxxxxxxxxxxxx", "tokenUrl": "https://xxxxxx.authentication.sap.hana.ondemand.com/oauth/token" }</pre> <p>For developer role:</p> <pre>{ "url": "https://xxxxxxxxxxxxx.cfapps.sap.hana.ondemand.com", "certificate": "xxxxxxxxxxxxx", "certurl":</pre>

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
				<pre> "https:// xxxxxx.authentication. cert.sap.hana. ondemand.com ", "clientId": "xxxxxxxxxxxxx xxxxxxxxxxxxx xxxxxxxxxxxxx x", "developerId": "xxxxxxxxxxxx xx", "privateKey": "xxxxxxxxxxxx xxxxxxxxxxxx xxx", "tokenUrl": "https:// xxxxxx.authentication. sap.hana.ondem and.com/ oauth/token" } </pre>

5. Click [Save](#).

The credentials like url, tokenUrl, developerId (for developer role), clientId, and clientSecret details are displayed for the given service key.

- The application url is used to make API calls.
- The clientId and clientSecret are necessary credentials required to fetch the Bearer Token.
- The tokenUrl is used to fetch the Bearer Token.

Make a note of these credentials as you will need them in the next steps to obtain a bearer token, in order to access the API business hub enterprise APIs.

Note

Once your client is setup you can use it to authenticate against the x509 endpoint by providing the client certificate and key.

Create the certificate.cer and certificate.key files based on the public and private keys obtained from the x509 credentials respectively.

For certificate.cer file:

1. Copy the "certificate" value starting from -----BEGIN CERTIFICATE----- all the way to -----END CERTIFICATE-----\n (the certificate value might contain multiple certificates). Make sure you copy all the certificates.

2. Paste it in a text editor. Find and replace all the occurrences of `\\n` by `\n`.
3. Save the file as `certificate.cer`.

For `certificate.key` file:

1. Copy the "certificate" value starting from `-----BEGIN RSA PRIVATE KEY-----\n...` all the way to `...-----END RSA PRIVATE KEY-----\n`
2. Paste it in a text editor. Find and replace all the occurrences of `\\n` by `\n`.
3. Save the file as `certificate.key`.

Open a command prompt/terminal in the folder where you have saved the certificate files and execute the following curl command to get the response in the `my-oauth-response.json` file in the same folder. From this file, you can fetch the bearer token from the value of "access_token".

```
curl --cert certificate.cer --key certificate.key --location --request
POST '<certurl from the servicekey x509 credentials>/oauth/token?
grant_type=client_credentials' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'client_id=<clientId from the servicekey x509 credentials>' \
--cert certificate.cer \
--key certificate.key \
> my-oauth-response.json
```

Updating a Service Instance in the API Management, API business hub enterprise

You can update an already provisioned service instance of an API access plan by performing the following steps:

Prerequisite:

You must have the Cloud Foundry CLI installed.

1. Log in to the Cloud Foundry CLI by running the `cf login` command.
2. Select *Org*.
3. Run the following command to update your service instance. `cf update-service <service-instance-name> -c <empty-json-file>.json`.

Sample Code

```
Sample json: {}
```

Next Steps

Obtaining a Bearer Token

In the REST client:

1. Paste the copied *tokenUrl*. Append `?grant_type=client_credentials` to the *tokenUrl*.

2. Choose `Basic Auth` as the `Authorization` type.
3. Similarly, paste the `clientId` and `clientSecret` in the place of `Username` and `Password`.
4. Make a POST Call.
5. Obtain the Bearer Token from the output and copy it in a notepad.
 - Now, to trigger an API, in the same REST client, append the API endpoint (obtained from the API business hub enterprise APIs that are located in the SAP API Management package of API Business Hub) to the `url`.
 - Choose `Bearer Token` as the `Authorization` type and paste the copied Bearer Token in the specified space.
 - Include payloads, if needed.
 - Make an API call.

Parent topic: [API Management Service Plans \[page 127\]](#)

Related Information

[Accessing API Management APIs Programmatically \[page 127\]](#)

[Managing Cloud Foundry Microservices through API Management \[page 147\]](#)

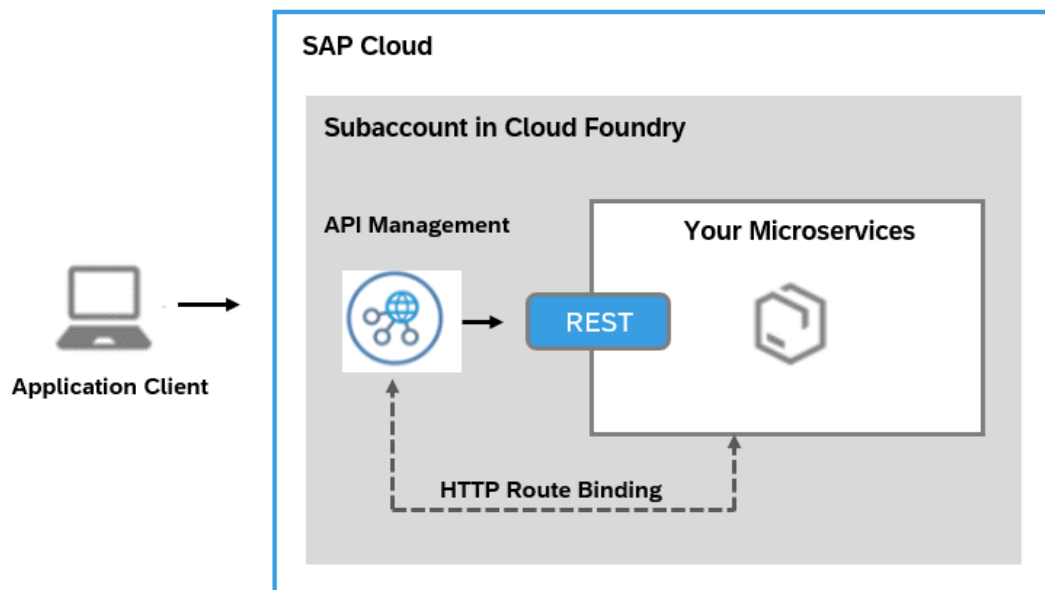
[Accessing On-Premise Systems through API Management \[page 150\]](#)

1.4.1.3.3 Managing Cloud Foundry Microservices through API Management

The `apim-as-route-service` plan helps you in managing Cloud Foundry applications by including policies such as rate limit, quota. The service instance you create through this plan allows you to bind to the route service and creates an API Proxy. This API Proxy serves in establishing a secure connection with your Cloud Foundry application and all the calls made to the Cloud Foundry application are routed via [API Management](#), [API portal](#).

About the Plan

This service plan is necessary if you have a microservice built and deployed on the SAP Cloud Foundry environment and you want to manage it using API Management. With this plan, even when calling the actual microservice URL (not the API proxified URL), the Cloud Foundry router will ensure that API Management is invoked first. This means that regardless of whether the API proxified URL or the actual microservice URL is called, API Management policies will always be enforced. This functionality is only available for APIs developed and deployed on the Cloud Foundry environment.



The service instance you create through this plan allows you to bind to the route service and creates an API Proxy. This API Proxy serves in establishing a secure connection with your Cloud Foundry application and all the calls made to the Cloud Foundry application are routed via *API Management, API portal*.

Prerequisites

- You've enabled API Management capability using Integration suite and have completed the setup. For more information, refer [Subscribing to Integration Suite](#) and [Activating Capabilities](#).
- You have the `space_developer` role assigned to you.

Creating an API Management, API portal Service Instance

Create a service instance in *API Management, API portal* to start managing your Cloud Foundry applications.

Follow the below procedure to create a service instance on Cloud Foundry:

1. In your web browser, open the *SAP BTP Cockpit* - <https://eu-access.cockpit.btp.cloud.sap>.
2. From your *Subaccount*, navigate to *Spaces* in your Cloud Foundry environment and choose **Services** **Service Marketplace**.
3. Choose **API Management, API portal** **Instances** **New Instance**.
4. In the *Create Instance* dialog, choose *apim-as-route-service* plan.
5. Choose *Next* until you reach the *Confirm* section.
6. In the section *Confirm*, enter a unique *Instance Name* and choose *Finish*.

Note

The *apim-as-route-service* plan allows you to create multiple service instances and connect to many Cloud Foundry applications using the same Subaccount.

Binding a Cloud Foundry Application to an API Management, API portal Service Instance

Create a service instance and bind the Cloud Foundry application to *API management, API portal* service. When you bind an application, an API proxy is created and a new route is added to the application. The route initially redirects all calls to the proxy URL and then to the application.

Prerequisites

- You have the `space developer` role assigned to you.
- You have created a service instance under *API Management, API portal*.

Open the command-line interface for Cloud Foundry and enter the following command:

Sample Code

```
cf bind-route-service sap-cf-domain.com apim-service-instance-name --hostname
my-app -c '{"api_name" : "custom_api_proxy_name"}'
<-- Example
//Without parameters
cf bind-route-service cfapps.sap.hana.ondemand.com apim-prod-instance --
hostname taxapp
//Cloud foundry URL for the above example is https://
taxapp.cfapps.sap.hana.ondemand.com
//With parameters for Linux/MAC system
cf bind-route-service sap-cf-domain.com apim-service-instance-name --hostname
my-app -c '{"api_name" : "test_api"}'
//With parameters for Windows system
cf bind-route-service sap-cf-domain.com apim-service-instance-name --hostname
my-app -c "{\"api_name\" : \"test_api\"}"
-->
```

Note

API Management, API portal supports only English alpha numerics, hyphens (-) and underscores (_) characters for "api_name".

You can bind an application to a service only from the command-line interface and not from SAP BTP Cockpit.

Providing a value for the parameter during binding is optional. If you provide a value for `api_name`, then the API proxy created in *API Management, API portal* for current binding gets the given name. Also, if an API with the same name exists in the API portal, then the same API proxy is used for the binding. That is, the API proxy end point is registered as the route service URL for the current binding.

Parent topic: [API Management Service Plans \[page 127\]](#)

Related Information

[Accessing API Management APIs Programmatically \[page 127\]](#)

[Accessing API business hub enterprise APIs Programmatically \[page 134\]](#)

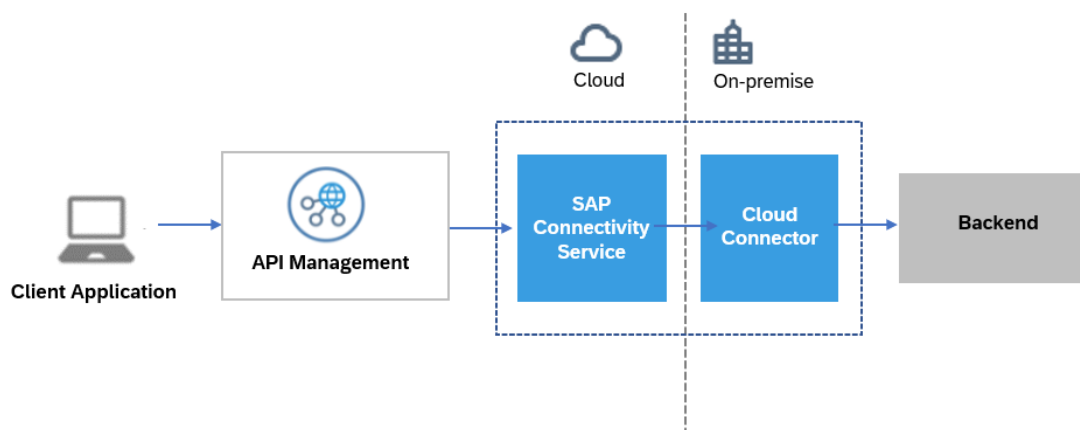
[Accessing On-Premise Systems through API Management \[page 150\]](#)

1.4.1.3.4 Accessing On-Premise Systems through API Management

The *on-premise-connectivity* plan helps in achieving principal propagation while connecting to an on-premise backend system.

About the Plan

Let us consider an use case where you want to pass the identity and security context of the logged-in user in the client application (known as the principal) from client application to on-premise backend. It ensures that the downstream services have the necessary information to authenticate the client without requiring the client to re-authenticate for each service. When a client makes a request to an API gateway, the gateway authenticates the user. It then propagates the principal information, such as the user's identity, to the backend services that the client's request needs to access. This allows the downstream services to make authorization decisions based on the user's details.



To accomplish principal propagation, you require a service key. This plan allows you to obtain the token by creating a service instance and generating a service key.

This topic explains how to obtain a service key in order to enable principal propagation using *API Management* in the Cloud Foundry Environment.

Prerequisites

- You've enabled API Management capability using Integration suite. For more information, refer [Subscribing to Integration Suite](#) and [Activating Capabilities](#).
- You have created an API Provider of type *On Premise* and chosen *Principal Propagation* as a mode of authentication to connect to an on-premise system. For more information, see [Create an API Provider \[page 538\]](#).
- You have the `space_developer` role assigned to you.

Creating a Service Instance on API Management, API Portal

Create a service instance to generate a service key that is used to enable the principal propagation.

1. In your web browser, open the *SAP BTP Cockpit* - <https://eu-access.cockpit.btp.cloud.sap>.
2. From your *Subaccount*, navigate to *Spaces* in your Cloud Foundry environment and choose **► Services** **► Service Marketplace**.
3. Choose **► API Management, API portal** **► Instances** **► New Instance**.
4. In the *Create Instance* dialog, choose *on-premise-connectivity* plan.
5. Click *Next* until you reach the *Confirm* section.
6. In the section *Confirm*, enter a unique *Instance Name*, and choose *Finish*.

Service Key Generation

After you have created a service instance, proceed with:

1. Choose the created service instance link from the visible list.
2. In the left-hand pane, navigate to **► Service Keys** **► Create Service Key**.
3. In the *Create Service* dialog, provide a *Name* and *Description* (optional).

4. In the text box enter one of the following payloads as per your requirement:

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
"instance-secret" (without payload)		Low	For instance-secret, the clientSecret generated is same for all the keys.	<pre> { "url": "https:// token- endpoint- url", "clientId": "your- client- id", "clientSecret": "xxxxxxxxxxxxx xxxxxxxxxxxxxx =", "orgId": "xxxxxxx- xxxx-xxxx- xxxx- xxxxxxxxxxx", "tokenUrl": "https:// token- endpoint-url/ oauth/token" } </pre>

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
"instance-secret" (with payload)	<pre>{ "xsuaa": { "credential-type": "instance-secret" } }</pre>	Low	For instance-secret, the clientSecret generated is same for all the keys.	<pre>{ "url": "https://token-endpoint-url", "clientId": "your-client-id", "clientSecret": "xxxxxxxxxxxx= ", "orgId": "xxxxxx-xxxx-xxxx-xxxx", "tokenUrl": "https://token-endpoint-url/oauth/token" }</pre>

To create service key of credential type...	Use payload...	Level of Security	Important Notes	Sample of generated credentials
"x509" (certificate based)	<pre>{ "xsuaa": { "credential-type": "x509" }, "x509": { "key-length": 2048 }, "validity": 65, "validity-type": "DAYS" }</pre>	High	For X509, ensure that the credential rotation is done based on the validity provided in the payload. For example, delete and create a new service key every 65 days.	For admin role: <pre>{ "url": "https://xxxxxx.authentication.sap.hana.ondemand.com", "certificate": "xxxxxxxxxxxx", "certurl": "https://xxxxxx.authentication.cert.sap.hana.ondemand.com", "clientId": "xxxxxxxxxxxx", "privateKey": "xxxxxxxxxxxx", "orgId": "xxxxxx-xxxx-xxxx-xxxx", "tokenUrl": "https://token-endpoint-url/oauth/token" }</pre>

- Choose [Save](#). The client credentials like [url](#), [clientId](#), [clientSecret](#), and [tokenUrl](#) details appear for the given instance name.

The [tokenUrl](#) is used to fetch the bearer token.

Example:

```
{
  "url": "https://<apiportal application name>.cfapps.sap.hana.ondemand.com",
  "tokenUrl": "https://<Space name>.authentication.sap.hana.ondemand.com/oauth/token",
  "clientId": "sb-opproxy-xsuaa-clonexxxxxxxxx!xxxxxx |opproxy-xsuaa!xxxxxx",
  "clientSecret": "xxxxxxxxxxxxxxxxxxxxxxxxxxxx="
}
```

```
}
```

Copy the client credentials in a notepad.

You can now navigate back to `Principal Propagation` from Neo to the Cloud Foundry environment or `Principal Propagation` from the same Cloud Foundry subaccount as you have the required client credentials.

You can use the credentials to establish:

- [Principal Propagation from the Neo to the Cloud Foundry Environment \[page 547\]](#): Enable an application in your subaccount in the Neo environment to access an API Management proxy created on a Cloud Foundry based *API Management*, *API portal* without a user login. For this scenario to work, the Neo subaccount needs to be trusted by the Cloud Foundry subaccount where *API Management*, *API portal* is enabled. Now, the application on Neo can call API Management proxy using OAuth2SAMLBearer destination.
- [Principal Propagation from the Same Cloud Foundry Subaccount \[page 550\]](#): Enable an application in your subaccount in the Cloud Foundry environment to access an API Management proxy created on the same Cloud Foundry based *API Management*, *API portal* without a user login. The JWT user token in your application can be exchanged with the API Management token using the service key credentials created for *API Management*, *API portal*.

Parent topic: [API Management Service Plans \[page 127\]](#)

Related Information

[Accessing API Management APIs Programmatically \[page 127\]](#)

[Accessing API business hub enterprise APIs Programmatically \[page 134\]](#)

[Managing Cloud Foundry Microservices through API Management \[page 147\]](#)

1.4.1.4 Consume API Management Service Instance from Kyma

Kyma environment provides a fully managed Kubernetes runtime based on the open-source project "Kyma". You can use the Kyma environment to search and discover API Portal and API business hub enterprise applications.

Prerequisites

- You already have a subaccount and have the required entitlements for *API Management*, *API Portal* and *API Management*, *API Business Hub Enterprise*.
- You've subscribed to *Integration Suite*, and have enabled *API Management*, *API Portal* and *API business hub enterprise* capability.

Note

If you're using the API Management stand-alone service, ensure that you've already subscribed to API Management, API portal and API business hub enterprise.

- Ensure that the Kyma environment has already been enabled. For more information, see [Getting Started in the Kyma Environment](#).

Note

You can also see the following tutorial for visual instructions to discover and consume APIs from Kyma: [Consume API Management Service Instance from Kyma](#).

Discover and Consume API Management Service Instance from Kyma

Before creating the service key, the **API Management, API Portal** and the **API Management, API Business Hub Enterprise** cluster services must be provisioned in the default namespace.

1. Log on to SAP BTP Cockpit.
2. Choose *Kyma Environment* tab and choose *Console URL: Link to dashboard* to navigate to the Kyma dashboard.
3. Choose *Namespaces* from the left navigation pane. At this point, only the *default* namespace appears.
4. From the left navigation pane, choose **Service Management** > *BTP Service Instances*.
5. Choose *Create Service Instance +* and on the *Create Service Instance* dialog fill in the following:
 - *Name*: Provide a name or generate it by choosing *Generate name*
 - *Offering Name*: You can find the SAP BTP service offering name in the *Service Marketplace* of the SAP BTP cockpit. Search for the API Management, API Portal/ API Management, API business hub enterprise service and copy the name of the service instance.
 - *Plan Name*: Similarly, you can find the relevant plan for the respective API Management, API Portal/ API Management, API business hub enterprise services in the *Service Marketplace* of the SAP BTP cockpit.
6. Choose *Advance* tab on the *Create Service Instance* dialog and expand *Instance Parameters*.
To create an instance of *API Management, API Portal*, enter **role** as the key in the first textbox and **APIPortal.Administrator** as the value in the second textbox.
Similarly, to create an instance of *API Management, API Business Hub Enterprise*, enter **role** in the first textbox and **AuthGroup.API.Admin** in the second textbox.
7. Choose *Create*.
You're directed to the **Service Management** > *Instances* page, where an instance of the newly provisioned *API Management, API Portal* or *API Management, API Business Hub Enterprise* service appears.

Creating the Service Key

To create the service key:

1. Navigate to the instance that you just created and choose *Add Service Binding +*.
2. Generate a *Name* for the service binding and select the *Service Instance Name* from the dropdown menu, and choose *Create*.
The service instance name, secret, and external name appears under *Binding Data* on the *Instances* page.
3. Select the link next to *Secret* and make a note of the *clientId*, *clientSecret orgId*, *tokenUrl* and the *url* details. You can use the *clientId*, *clientSecret* and the *tokenUrl* to fetch the token, and the application URL to perform operation on the APIs.

1.4.1.5 Account Members

All members in your enterprise who need to use API Management applications need to be assigned to the account.

Members can use the application within the scope of the account and based on the roles assigned to the members. For information on how to assign members to the account, see [Managing Members](#).

1.4.1.6 User Roles in API Management

Use role collections to group together different roles that can be assigned to API Portal and API business hub enterprise users.

API Portal Roles

Role Collection	Description
<code>APIPortal.Administrator</code>	Use this role to access the API portal user interface (UI) and services, manage the API proxies by adding additional policies, and create products. You can also use this role to manage APIs using the API Designer.
<code>APIPortal.Service.CatalogIntegration</code>	Use this role to establish a connection from the API business hub enterprise to the API portal.
<code>APIManagement.Selfservice.Administrator</code>	Use this role during the onboarding of API Portal and to get access to its settings page.
<code>APIPortal.Guest</code>	Use this role to access the API portal in read-only mode. You can view all APIs, policies, API providers, and analytics, but can't edit them.

API business hub enterprise Roles

Role Collection	Description
<code>AuthGroup.SelfService.Admin</code>	Use this role during the onboarding of API business hub enterprise and to get access to it.

Role Collection	Description
AuthGroup.API.Admin	<p>Use this role to:</p> <ul style="list-style-type: none"> • Manage an application developer's access to the portal by either accepting or rejecting an application developer's request. In addition, you can revoke the access of an existing application developer. • Manage roles for a user by adding new roles or removing existing roles. • On-behalf of an application developer, admin can also perform the following tasks: <ul style="list-style-type: none"> • Create, update, and delete applications. • Create custom attributes for applications. • Provide app key and secret, while creating or updating an application.
AuthGroup.ContentAuthor	<p>Use this role to:</p> <ul style="list-style-type: none"> • Publish content to the API business hub enterprise. • Establish a connection from the API portal to the API business hub enterprise.
AuthGroup.API.ApplicationDeveloper	<p>Use this role to:</p> <ul style="list-style-type: none"> • Access the API business hub enterprise. • Create, update, and delete applications. • View analytics information on application usage, performance, and error count. • View and download bills for subscribed applications.
AuthGroup.Content.Admin	<p>Use this role to:</p> <ul style="list-style-type: none"> • Create and update categories.
AuthGroup.Site.Admin	<p>Use this role to:</p> <ul style="list-style-type: none"> • Configure updates. • Perform portal changes like uploading logo, changing the name and description, and changing the footer links.

1.4.1.6.1 Assigning Role Collections to Users

Role collections enable you to group together the roles you create. The role collections you define can be assigned to users logged on to SAP ID service.

Procedure

1. In your web browser, open [SAP BTP Cockpit](#) and choose the relevant subaccount.
2. In the left-hand pane, choose ► [Security](#) ► [Role Collections](#) ►
3. Choose the role collection to which you want to assign users.
4. Go to the [Users](#) section and choose [Edit](#).
5. Enter the user ID of the user that you want to assign to the role collection. If the user only exists in a connected identity provider, you must choose the identity provider and type in the e-mail address.
6. (Optional) To add more users, choose + (Add a user).
7. Save your changes.

Related Information

[Set Up API Portal Application \[page 120\]](#)

[Set Up API business hub enterprise Application Using the Standalone Tile \[page 125\]](#)

[Shadow Users \[page 176\]](#)

1.4.1.6.2 Creating a Custom Role

Create a custom role for API products in API Management.

Context

You can restrict access to an API product in API Management using a custom role. That is, only an authorized user can discover and subscribe to API Products that are tagged to a custom role.

📘 Note

If you're using Integration Suite, refer [Create Roles for Applications Using Existing Role Templates](#) to create a custom role for API Products.

To create a custom role for API Product, use the [ApplicationDeveloper](#) role template. Also, ensure that for the [CustomRole](#) attribute, you choose the value of [Source](#) as [Static](#), and in the [Values](#), specify the attribute values and press enter. This value is later used to assign permission while creating an API Product.

Procedure

1. Go to your *Subaccount* in *SAP BTP Cockpit* for Cloud Foundry environment.
2. Choose *Service Marketplace* in the left-hand pane.
3. In order to create a custom role, choose the *API Management, API Business Hub Enterprise* tile.
4. Under *Application Plans* for API BusinessHub Enterprise, choose the **••• Action** icon and select *Manage Roles*.
5. To add a new custom role, choose **+ Add a role**.
6. In the *Create Role* dialog, fill the details for:
 - Role Name
 - Description
 - Role Template: Choose *ApplicationDeveloper*.
 - Attributes: For the *CustomRole* attribute, keep the value of *Source* as *static*. Under *Values* specify the attribute values and press enter.

Note

Use only alphanumeric characters for the attribute values. Also, the attribute values shouldn't contain any spaces or special characters except for the hyphen '-' and underscore '_'.


It is recommended that you use CamelCase for better readability.

The *Next* option on the *Create Role* dialogue will remain greyed out until you press enter after typing the attribute values in the text box.

Attributes	Source	Values
CustomRole	Static	ReadOnlyRole

This value is later used to assign permission while creating an API product.
A new role is created and added to the Roles list.

7. **Add the created role to Role Collection:** Adding a custom role to the role collection ensures that you choose a specific application and role template.

- a. Navigate to your **Subaccount** > **Security** > **Role Collections** . Choose **+ Create New Role Collection** and provide a *Name* and *Description* to the new role collection.
 - b. Choose the newly created *Role Collection* and choose *Edit*.
 - c. To select the custom role, choose the  icon under the *Roles* tab.
 - d. On the *Select: Role* dialog, choose the custom role from the *Role Name* dropdown, select the checkbox under *Roles*, and choose *Add*.
 - e. Choose *Save*.
8. **Assign role collection to the user:** To assign the created role collection to your authorized email ID:
1. Go to the *Users* section and choose *Edit*.
 2. Enter the user ID of the user that you want to assign to the role collection. If the user only exists in a connected identity provider, you must choose the identity provider and type in the email address.
 3. (Optional) To add more users, choose **+** (Add a user).
 4. Save your changes.

→ Remember

Application Developers who are already onboarded in the API business hub enterprise should have the custom role. If any user has been assigned a custom role but hasn't been onboarded as an application developer in the API business hub enterprise, the application creation fails. In this case, Authgroup.API.Admin can onboard the user as an Application Developer in the portal.

Next Steps

After completing the above steps, assign permissions while creating a Product in API Portal application. For more information on the same refer, [here \[page 658\]](#).

Related Information

[Create a Product \[page 653\]](#)

1.4.2 Custom Domain Configuration for API Portal or API business hub enterprise Subscription

To complete the process of configuring a custom domain for the API portal or the API business hub enterprise application using the Custom Domain Service in the SAP BTP Cloud Foundry environment, you need to contact the SAP API Management operations team.

Note

Custom domain for subscription URLs is available for *API Management*, *API portal*, *API Management*, *API business hub enterprise*, and API Management as a capability within the Integration Suite product.

For information on the initial setup of the Custom Domain Service in the Cloud Foundry environment, see [Configuring Custom Domains](#).

To map the custom domain to a SAAS application subscription, see [Map a SaaS Application to a Custom Domain](#).

After mapping a custom domain to a SAAS application subscription using the Custom Domain Service, you need to raise a ticket through the [SAP Support Portal](#) to complete the SaaS route configuration.

Use the following component for your incident:

Component Name	Component Description
OPU-API-OD-OPS	SAP API Management Operations - On Demand

When submitting the incident, include the following information:

- Subdomain of the current account
- Application: API Portal/ API business hub enterprise
- Custom domain URL (the host name for the URL)
- API Portal/API business hub enterprise URL

1.4.3 Region-Specific IP Addresses Available for API Management Cloud Foundry Environment

API Management protects your backend services. However, API Management needs to establish connectivity to your backend services during an API call execution.

In case your backend service is restricting access to certain IPs as part of security measures, you need to add API Management NAT IPs to the list of allowed IPs in your backend services.

Note

- NAT (Network Address Translation) IPs are egress IPs for requests from API Management.
- In the Cloud Foundry environment, IPs are controlled by the respective IaaS provider (AWS, Azure, Google Cloud, Alibaba Cloud). IPs may change due to network updates on the provider side. Any planned changes will be announced at least four weeks before they take effect.

To get region-specific egress (outbound) NAT IP addresses for API Management, see the following table:

Regions for Enterprise Accounts

IaaS Provider	Region	Region Name	Technical Key	Technical Key of IaaS Provider	NAT IPs (egress, for outgoing requests)
Microsoft Azure	eu20	Europe (Netherlands)	cf-eu20	West Europe	51.105.226.79, 20.107.78.224, 20.4.205.181
Microsoft Azure	ap20	Australia (Sydney)	cf-ap20	Australia East	20.53.178.190
Microsoft Azure	ap21	Singapore	cf-ap21	Southeast Asia	20.43.177.113

IaaS Provider	Region	Region Name	Technical Key	Technical Key of IaaS Provider	NAT IPs (egress, for outgoing requests)
Microsoft Azure	us20	US West (WA)	cf-us20	West US 2	51.143.126.237
Microsoft Azure	jp20	Japan (Tokyo)	cf-jp20	Japan East	52.155.117.53
Microsoft Azure	us21	US East (VA)	cf-us21	East US	20.42.28.32
Amazon Web Services	br10	Brazil (São Paulo)	cf-br10	sa-east-1	18.229.180.216, 18.230.68.32, 18.229.200.51
Amazon Web Services	jp10	Japan (Tokyo)	cf-jp10	ap-northeast-1	52.69.140.122, 18.181.69.241, 18.182.245.202
Amazon Web Services	ap10	Australia (Sydney)	cf-ap10	ap-southeast-2	3.105.155.212, 13.211.74.25, 13.55.87.26
Amazon Web Services	ap11	Asia Pacific (Singapore)	cf-ap11	ap-southeast-1	54.254.127.94, 54.179.36.212, 54.151.195.2
Amazon Web Services	ap12	Asia Pacific (Seoul)	cf-ap12	ap-northeast-2	3.35.108.250, 54.180.45.228, 3.36.176.209
Amazon Web Services	ca10	Canada (Montreal)	cf-ca10	ca-central-1	3.96.232.61, 3.96.230.37, 15.222.204.174
Amazon Web Services	eu10	Europe (Frankfurt)	cf-eu10	eu-central-1	52.29.48.148, 3.120.95.10, 18.194.144.165, 18.196.191.48, 52.59.78.206, 18.195.138.5, 3.73.160.117, 52.57.130.124, 3.72.189.179
Amazon Web Services	eu11	Europe (Frankfurt)	cf-eu11	eu-central-1	18.156.85.8, 3.65.144.116, 3.121.107.212
Amazon Web Services	us10	US East (VA)	cf-us10	us-east-1	3.213.79.219, 3.209.244.202, 3.213.81.148, 54.87.110.53, 54.208.172.140, 54.86.163.159
Google Cloud	us30	US Central (IA)	cf-us30	us-central-1	35.223.165.172, 34.133.13.45, 34.72.36.170
Alibaba Cloud	cn40	China (Shanghai)	cf-cn40	cn-shanghai	101.132.190.155, 106.14.165.33, 106.14.184.113
Microsoft Azure	ch20	Switzerland (Zurich)	cf-ch20	Switzerland North	20.250.216.117, 20.250.176.24

IaaS Provider	Region	Region Name	Technical Key	Technical Key of IaaS Provider	NAT IPs (egress, for outgoing requests)
Google Cloud	in30	India (Mumbai) GCP	cf-in30	asia-south1	34.100.176.82, 34.93.181.26, 35.200.251.32, 34.100.182.244, 34.93.184.71, 34.100.176.113
Google Cloud	eu30	Europe (Frankfurt) GCP	cf-eu30	europa-west3	34.159.208.178, 34.159.31.39, 34.141.20.163, 34.107.78.67, 34.159.45.83, 35.246.220.63

Regions for Trial Accounts

IaaS Provider	Region	Region Name	Technical Key	Technical Key of IaaS Provider	NAT IPs (egress, for outgoing requests)
Microsoft Azure	ap21	Singapore	cf-ap21	Southeast Asia	20.195.52.254
Amazon Web Services	eu10	Europe (Frankfurt)	cf-eu10	eu-central-1	18.157.223.60, 18.157.143.164, 52.28.147.96
Amazon Web Services	us10	US East (VA)	cf-us10	us-east-1	54.172.191.202, 52.175.180, 52.207.193.169

Note

If customers are implementing allow or deny listing based on IPs from their clients to API Management design time applications - API Portal, API Business Hub Enterprise, or other platform services in Cloud Foundry, such as SAP Authorization and Trust Management Service (XSUAA) for fetching tokens, they will need to refer to the documentation for SAP Business Technology Platform Cloud Foundry IP addresses, which can be found at the following link: [Regions and API Endpoints Available for the Cloud Foundry Environment | SAP Help Portal](#).

1.4.4 Configuring Additional Virtual Host in Cloud Foundry Environment

A virtual host allows you to host multiple domain names on the API Management capability within Integration Suite.

Create a new virtual host with default domain or custom domain and update alias, keystore, keyalias, and truststore for an existing virtual host in the Cloud Foundry environment.

[Configuring a Default Domain for a Virtual Host \[page 166\]](#)

After successful onboarding, API proxies are assigned a default virtual host URL. Currently, this URL uses the domain "ondemand.com," which is the common domain for the Business Technology Platform. It's prefixed with the subdomain consisting of the subaccount name and the data center where the Integration Suite tenant is onboarded. For example, the default host alias could be `https://myaccount...eu10.hana.ondemand.com`.

[Configuring a Custom Domain for a Virtual Host \[page 170\]](#)

The API Management capability enables you to personalize the virtual host URL by configuring a custom domain of your choice. This means that you can have all your APIs displayed as "https://api.bestrun.com/..." if desired. Additionally, you have the option to set up multiple virtual hosts using the same custom domain, such as "https://api1.bestrun.com," "https://api2.bestrun.com," and so on.

[Configuring Mutual TLS for Virtual Host \[page 174\]](#)

You can configure mutual TLS for a virtual host, which validates the identities of both the web server and the web client.

1.4.4.1 Configuring a Default Domain for a Virtual Host

After successful onboarding, API proxies are assigned a default virtual host URL. Currently, this URL uses the domain "ondemand.com," which is the common domain for the Business Technology Platform. It's prefixed with the subdomain consisting of the subaccount name and the data center where the Integration Suite tenant is onboarded. For example, the default host alias could be `https://myaccount...eu10.hana.ondemand.com`.

Prerequisites

- You must have a service key for the `APIManagement.SelfService.Administrator` role. To know more about creating a service key for accessing APIs in the API portal, see the **Creating a Service Key** section in [Accessing API Management APIs Programmatically \[page 127\]](#).
- You have fetched a valid bearer token. To know more about obtaining a bearer token, see the **Obtaining a Bearer Token** section in [Accessing API Management APIs Programmatically \[page 127\]](#).

Context

If you want to configure a mutual TLS, see [Configuring Mutual TLS for Virtual Host \[page 174\]](#). This process enables you to establish a more secure connection between your API Proxy and the client, ensuring that both parties can trust each other's identities.

To request a custom domain with one-way TLS, perform the following steps:

Procedure

1. Run the services using a standard REST console.
2. Service to create a new virtual host:
 - Service URL: `https://<url-from-service-key>/apiportal/operations/1.0/Configuration.svc/VirtualHostRequests`
 - Method: POST
 - Request Header: Authorization: Bearer <token>
 - Content Type: application/json
 - Accept: application/json
 - Request Body:

Sample Code

```
{
  "accountId" : "subdomain of your subaccount",
  "virtualHostUrl" : "prod-apis",
  "isDefaultVirtualHostRequest" : false,
  "operation" : "CREATE"
}
```

Note

- accountId: This is the subdomain of your subaccount.
- virtualHostUrl - This is your virtual host alias, for example, prod-apis, testapi.
- isDefaultVirtualHostRequest -if you want the new virtual host to be the default virtual host, set the value to "true", else set it to "false".

Note

To enable client authentication (mutual TLS) while configuring the virtual host with default domain, see [Configuring Mutual TLS for Virtual Host \[page 174\]](#).

- Response: 201

Sample Code

```
{
  "d": {
    "__metadata": {
      "id": "https://apiportalurl:443/apiportal/api/1.0/Management.svc/VirtualHosts('1F02AD6A-A53C-43F4-BF95-F053A8A1469A')",
      "uri": "https://apiportalurl:443/apiportal/api/1.0/Management.svc/VirtualHosts('1F02AD6A-A53C-43F4-BF95-F053A8A1469A')",
      "type": "apimgmtconfiguration.VirtualHostRequest"
    },
    "accountId": "subdomain of your subaccount",
    "allocatedPort": 443,
    "allocationStatus": "COMPLETE",
    "clusterName": "",
    "id": "1F02AD6A-A53C-43F4-BF95-F053A8A1469A",
    "isClientAuthEnabled": false,
    "isDefaultVirtualHostRequest": false,
    "isForCustomDomain": false,
  }
}
```

```

        "isForNonSni": false,
        "isTLS": false,
        "keyStoreAlias": null,
        "keyStoreName": null,
        "life_cycle": {
            "__metadata": {
                "type": "apimgmtconfiguration.History"
            },
            "changed_at": "/Date(1707380514905)/",
            "changed_by": "sb-apiaccess_1705298659201!b109482|api-portal-xsuaa!b11864",
            "created_at": "/Date(1707380504072)/",
            "created_by": "sb-apiaccess_1705298659201!b109482|api-portal-xsuaa!b11864"
        },
        "operation": "CREATE",
        "trustStore": null,
        "virtualHostId": "c269915f-7adc-4f78-bdd0-dd39ffcb079f",
        "virtualHostUrl": "prod-apis.sapdefaultdomain",
        "lbHost": null
    }
}

```

3. Service to update a virtual host:

You can update the virtualHostUrl, trustStore, and the isDefaultVirtualHostRequest flag.

You can also convert your default domain virtual host to custom domain by referring to the update section (Step 3) of the [Configuring a Custom Domain for a Virtual Host \[page 170\]](#) topic.

- Service URL: `https://<url-from-service-key>/apiportal/operations/1.0/Configuration.svc/VirtualHostRequests`
- Method: POST
- Request Header: Authentication Bearer <token>
- Content Type: application/json
- Request Body:

Sample Code

```

{
    "accountId" : "subdomain of your subaccount",
    "virtualHostUrl": "prod-apis-updated",
    "isDefaultVirtualHostRequest" : false,
    "operation" : "UPDATE",
    "virtualHostId": "c269915f-7adc-4f78-bdd0-dd39ffcb079f"
}
<!--
-->

```

Note

- accountId: This is the subdomain of your subaccount.
- virtualHostUrl - This is your virtual host alias, for example, prod-apis, testapi.
- isDefaultVirtualHostRequest -if you want the new virtual host to be the default virtual host, set the value to "true", else set it to "false".
- virtualHostId: This is the unique ID of the virtual host you are trying to update.

Note

To enable client authentication (mutual TLS) while configuring the virtual host with default domain, see [Configuring Mutual TLS for Virtual Host \[page 174\]](#).

- Response: 201

Sample Code

```
{
  "d": {
    "__metadata": {
      "id": "https://apiportalurl:443/apiportal/api/1.0/Management.svc/VirtualHostRequests('2F02AD6A-A53C-43F4-BF95-F053A8A1469B')",
      "uri": "https://apiportalurl:443/apiportal/api/1.0/Management.svc/VirtualHostRequests('2F02AD6A-A53C-43F4-BF95-F053A8A1469B')",
      "type": "apimgmtconfiguration.VirtualHostRequest"
    },
    "accountId": "subdomain of your subaccount",
    "allocatedPort": 443,
    "allocationStatus": "COMPLETE",
    "clusterName": "",
    "id": "2F02AD6A-A53C-43F4-BF95-F053A8A1469B",
    "isClientAuthEnabled": false,
    "isDefaultVirtualHostRequest": false,
    "isForCustomDomain": false,
    "isForNonSni": false,
    "isTLS": false,
    "keyStoreAlias": null,
    "keyStoreName": null,
    "life_cycle": {
      "__metadata": {
        "type": "apimgmtconfiguration.History"
      },
      "changed_at": "/Date(1707380514905)/",
      "changed_by": "sb-apiaccess_1705298659201!b109482|api-portal-xsuaa!b11864",
      "created_at": "/Date(1707380504072)/",
      "created_by": "sb-apiaccess_1705298659201!b109482|api-portal-xsuaa!b11864"
    },
    "operation": "UPDATE",
    "trustStore": null,
    "virtualHostId": "c269915f-7adc-4f78-bdd0-dd39ffcb079f",
    "virtualHostUrl": "prod-apis-updated.sapdefaultdomain",
    "lbHost": null
  }
}
```

- Response: 201

Note

After the virtual host is updated, APIs associated to a product using the updated virtual host must be redeployed and republished.

Task overview: [Configuring Additional Virtual Host in Cloud Foundry Environment \[page 124\]](#)

Related Information

[Configuring a Custom Domain for a Virtual Host \[page 170\]](#)

[Configuring Mutual TLs for Virtual Host \[page 174\]](#)

1.4.4.2 Configuring a Custom Domain for a Virtual Host

The API Management capability enables you to personalize the virtual host URL by configuring a custom domain of your choice. This means that you can have all your APIs displayed as "https://api.bestrun.com/..." if desired. Additionally, you have the option to set up multiple virtual hosts using the same custom domain, such as "https://api1.bestrun.com," "https://api2.bestrun.com," and so on.

Prerequisites

- You must have a service key for the **APIManagement.SelfService.Administrator** role. To know more about creating a service key for accessing APIs in the API portal, see the **Creating a Service Key** section in [Accessing API Management APIs Programmatically \[page 127\]](#).
- You have fetched a valid bearer token. To know more about obtaining a bearer token, see the **Obtaining a Bearer Token** section in [Accessing API Management APIs Programmatically \[page 127\]](#).

Context

The advantage of establishing your own custom domain with a virtual host is that it protects you from future changes, such as those that may occur when rehosting your APIs.

In addition, you have the option to secure your custom domain with a default one-way TLS or a mutual TLS. A one-way TLS validates only the secure identity of the API Proxy, providing encryption and authentication for the server. On the other hand, a mutual TLS validates the identities of both the API Proxy and the client, providing mutual authentication.

If you want to configure a mutual TLS, see [Configuring Mutual TLs for Virtual Host \[page 174\]](#). This process enables you to establish a more secure connection between your API Proxy and the client, ensuring that both parties can trust each other's identities.

To request a custom domain with one-way TLS, perform the following steps:

Procedure

1. Run the services using a standard REST console.
2. Service to create a new virtual host:

- Service URL: `https://<url-from-service-key>/apiportal/operations/1.0/Configuration.svc/VirtualHostRequests`
- Method: POST
- Request Header: Authorization: Bearer <token>
- Content Type: application/json
- Accept: application/json
- Request Body:

↔ Sample Code

```
{
  "accountId" : "subdomain of your subaccount",
  "virtualHostUrl" : "apis.customdomain.com",
  "isDefaultVirtualHostRequest" : false,
  "isForCustomDomain": true,
  "keyStoreName": "ref://<reference_name>" or "<key_store_name>",
  "keyStoreAlias": "key_store_alias",
  "operation" : "CREATE"
}
```

📌 Note

- `accountId`: This is the subdomain of your subaccount.
- `virtualHostUrl` - This is your virtual host URL which is used to access the deployed API proxies, e.g `prod-apis.customdomain.com`, `testapi.customdomain.com`.
- `isDefaultVirtualHostRequest` -If you want the new virtual host to be the default virtual host, set the value to "true", else set it to "false".
- `isForCustomDomain` - Ensure that the value is set to "true".
- `keyStoreName`: The `keyStoreName` parameter refers to the name of the keystore that should contain the custom domain's public and private key, or the name of the certificate store reference pointing to the keystore. To learn how to create a keystore and upload certificates, see [Manage Certificates \[page 558\]](#). Alternatively, you can use a certificate store reference name that points to the keystore containing the custom domain's public and private keys. For more information, see [Working with References \[page 560\]](#).
- `keyStoreAlias`: The `keyStoreAlias` parameter refers to the name of the keystore certificate containing the custom domain's public and private key. To learn how to create a keystore certificate and upload certificates, see [Manage Certificates \[page 558\]](#).

📌 Note

To enable client authentication (mutual TLS) while configuring the virtual host with custom domain, see [Configuring Mutual Tls for Virtual Host \[page 174\]](#).

- Response: 201

↔ Sample Code

```
{
  "d": {
    "__metadata": {
      "id": "https://apiportalurl:443/apiportal/api/1.0/Management.svc/VirtualHostRequest('1F02AD6A-A53C-43F4-BF95-F053A8A1469A')",

```

```

"uri": "https://apiportalurl:443/apiportal/api/1.0/Management.svc/
VirtualHostRequest('1F02AD6A-A53C-43F4-BF95-F053A8A1469A')",
  "type": "apimgmtconfiguration.VirtualHostRequest"
},
"accountId": "subdomain of your subaccount",
"allocatedPort": 443,
"allocationStatus": "COMPLETE",
"clusterName": "",
"id": "1F02AD6A-A53C-43F4-BF95-F053A8A1469A",
"isClientAuthEnabled": false,
"isDefaultVirtualHostRequest": false,
"isForCustomDomain": true,
"isForNonSni": false,
"isTLS": false,
"keyStoreAlias": "key_store_alias",
"keyStoreName": "ref://<reference_name> or "<key_store_name>",
"life_cycle": {
  "__metadata": {
    "type": "apimgmtconfiguration.History"
  },
  "changed_at": "/Date(1707380514905)/",
  "changed_by": "sb-apiaccess_1705298659201!b109482|api-
portal-xsuaa!b11864",
  "created_at": "/Date(1707380504072)/",
  "created_by": "sb-apiaccess_1705298659201!b109482|api-
portal-xsuaa!b11864"
},
"operation": "CREATE",
"trustStore": null,
"virtualHostId": "c269915f-7adc-4f78-bdd0-dd39ffcb079f",
"virtualHostUrl": "apis.customdomain.com",
"lbHost": "lbHost url"
}
}

```

The "lbHost" field contains the host URL which is required for the custom domain DNS mapping.

3. Service to update a virtual host:

You can update the virtualHostUrl, keyStoreName, keyStoreAlias, trustStore, and the isDefaultVirtualHostRequest flag

- Service URL: <https://<url-from-service-key>/apiportal/operations/1.0/Configuration.svc/VirtualHostRequests>
- Method: POST
- Request Header: Authentication Bearer <token>
- Content Type: application/json
- Accept: application/json
- Request Body:

Sample Code

```

{
  "accountId": "subdomain of your subaccount",
  "virtualHostUrl": "apisupdated.customdomain.com",
  "isDefaultVirtualHostRequest": false,
  "isForCustomDomain": true,
  "keyStoreName": "ref://<reference_name> or "<key_store_name>",
  "keyStoreAlias": "key_store_alias",
  "operation": "UPDATE",
  "virtualHostId": "1F02AD6A-A53C-43F4-BF95-F053A8A1469A"
}

```

```
}
```

Note

- `accountId`: This is the subdomain of your subaccount.
- `virtualHostUrl` - This is your virtual host URL which is used to access the deployed API proxies, e.g `prod-apis.customdomain.com`, `testapi.customdomain.com`.
- `isDefaultVirtualHostRequest` -If you want the new virtual host to be the default virtual host, set the value to "true", else set it to "false".
- `isForCustomDomain` - Ensure that the value is set to "true".
- `keyStoreName`: The `keyStoreName` parameter refers to the name of the keystore that should contain the custom domain's public and private key, or the name of the certificate store reference pointing to the keystore. To learn how to create a keystore and upload certificates, see [Manage Certificates \[page 558\]](#). Alternatively, you can use a certificate store reference name that points to the keystore containing the custom domain's public and private keys. For more information, see [Working with References \[page 560\]](#).
- `keyStoreAlias`: The `keyStoreAlias` parameter refers to the name of the keystore certificate containing the custom domain's public and private key. To learn how to create a keystore certificate and upload certificates, see [Manage Certificates \[page 558\]](#).
- `virtualHostId`: This is the unique ID of the virtual host you are trying to update.

Note

To enable client authentication (mutual TLS) while configuring the virtual host with custom domain, see [Configuring Mutual Tls for Virtual Host \[page 174\]](#).

- Response: 201

Sample Code

```
{
  "d": {
    "__metadata": {
      "id": "https://apiportalurl:443/apiportal/api/1.0/
Management.svc/VirtualHosts('689333a2-2f6b-4029-8734-2b36a687e559')",
      "uri": "https://apiportalurl:443/apiportal/api/1.0/
Management.svc/VirtualHosts('689333a2-2f6b-4029-8734-2b36a687e559')",
      "type": "apimgmtconfiguration.VirtualHostRequest"
    },
    "accountId": "subdomain of your subaccount",
    "allocatedPort": 443,
    "allocationStatus": "COMPLETE",
    "clusterName": "",
    "id": "1F02AD6A-A53C-43F4-BF95-F053A8A1469A",
    "isClientAuthEnabled": false,
    "isDefaultVirtualHostRequest": false,
    "isForCustomDomain": true,
    "isForNonSni": false,
    "isTLS": false,
    "keyStoreAlias": "key_store_alias",
    "keyStoreName": "ref://<reference_name> or "<key_store_name>",
    "life_cycle": {
      "__metadata": {
        "type": "apimgmtconfiguration.History"
      },
      "changed_at": "/Date(1707380514905)/",

```

```
        "changed_by": "sb-apiaccess_1705298659201!b109482|api-portal-xsuaa!b11864",
        "created_at": "/Date(1707380504072)/",
        "created_by": "sb-apiaccess_1705298659201!b109482|api-portal-xsuaa!b11864"
    },
    "operation": "UPDATE",
    "trustStore": null,
    "virtualHostId": "c269915f-7adc-4f78-bdd0-dd39ffcb079f",
    "virtualHostUrl": "apisupdated",
    "lbHost": "lbHost url"
}
}
```

Note

The "lbHost" field contains the host URL that is required for the custom domain DNS mapping. If the "lbHost" field does not display any value, please raise a support ticket through the [SAP Support Portal](#) using the component OPU-API-OD-OPS.

Note

After the virtual host is updated, APIs associated to a product using the updated virtual host must be redeployed and republished.

Task overview: [Configuring Additional Virtual Host in Cloud Foundry Environment \[page 124\]](#)

Related Information

[Configuring a Default Domain for a Virtual Host \[page 166\]](#)

[Configuring Mutual TLs for Virtual Host \[page 174\]](#)

[Configuring Additional Virtual Host in Cloud Foundry Environment \[page 124\]](#)

1.4.4.3 Configuring Mutual TLs for Virtual Host

You can configure mutual TLs for a virtual host, which validates the identities of both the web server and the web client.

Context

Note

Since client certificate chains are used in the authentication process to establish the identity of clients accessing the API Management service, it is important to ensure that these chains have sufficient

security measures in place. Weak client certificate chains lack the necessary security measures and are therefore vulnerable to attacks. As a result, weak client certificate chains have been deprecated. For more detailed information, please [3418201 - Deprecation of Weak Client Certificate Chains in API Management \(sap.corp\)](#).

Procedure

1. Run the services using a standard REST console.
2. Service to configure client authentication for a new or existing virtual host:
 - Service URL: `https://<url-from-service-key>/apiportal/operations/1.0/Configuration.svc/VirtualHostRequests`
 - Method: POST
 - Request Header: Authentication Bearer <token>
 - Content Type: application/json
 - Accept: application/json
 - Request Body:
Create

Sample Code

```
{
  "accountId" : "subdomain of your subaccount",
  "virtualHostUrl" : "prod-apis",
  "isDefaultVirtualHostRequest" : false,
  "isClientAuthEnabled" : true,
  "trustStore" : "ref://<reference_name>" or "<trust_store_name>",
  "operation" : "CREATE"
}
```

Update

Sample Code

```
{
  "accountId" : "subdomain of your subaccount",
  "virtualHostUrl" : "prod-apis",
  "isDefaultVirtualHostRequest" : false,
  "isClientAuthEnabled" : true,
  "trustStore" : "ref://<reference_name>" or "<trust_store_name>",
  "virtualHostId" : "c269915f-7adc-4f78-bdd0-dd39ffcb079f",
  "operation" : "UPDATE"
}
```

Note

- `isClientAuthEnabled`: This field must be set to "true" to enable mutual TLS.
- `trustStore`: This refers to the name of the truststore that holds the client certificate, or name of the certificate store reference that points to the trust store. To learn how to create a truststore and upload certificates, see [Manage Certificates \[page 558\]](#). Alternatively, you can use a certificate store reference name instead of the truststore name. This reference name points to

the truststore that contains the client certificate. For detailed instructions, see [Working with References \[page 560\]](#).

ⓘ Note

For enabling client authentication (mutual TLS) for custom domain virtual host, append `isClientAuthEnabled` and `trustStore` fields to the create/update virtual host request.

ⓘ Note

After the virtual host is updated, APIs associated to a product using the updated virtual host must be redeployed and republished.

Task overview: [Configuring Additional Virtual Host in Cloud Foundry Environment \[page 124\]](#)

Related Information

[Configuring a Default Domain for a Virtual Host \[page 166\]](#)

[Configuring a Custom Domain for a Virtual Host \[page 170\]](#)

1.4.5 Shadow Users

Whenever a user authenticates at an application in your subaccount using any identity provider, it's essential that user-related data provided by the identity provider is stored in the form of shadow users.

Previously, the user account and authentication service allowed any user of any connected identity provider to authenticate to applications in the subaccount. When there was no corresponding shadow user, it automatically created one based on the information received from the identity provider.

With new subaccounts created after 24 September 2020, automatic creation of shadow users is switched off by default for the default identity provider, SAP ID service. You can enable or disable automatic shadow user creation using the information [here](#).

Since automatic shadow user creation is disabled, if you've not explicitly created shadow users for your developers, then they're unable to log on to the application, and they're asked to contact the administrator.

You can create the shadow users for your developers in the SAP BTP cockpit, typically when you assign the first role collection to them. For more information on role collection assignment, see [Assigning Role Collections to Users or User Groups](#).

For information on creating shadow accounts, see [Add Users to SAP ID Service in the Cloud Foundry Environment](#).

You can also use the [User Management \(System for Cross-domain Identity Management \(SCIM\)\) API](#)  to manage your shadow users.

1.4.6 Cancel API Management Service Subscription

You can deactivate your API Management capability from Integration Suite to disable your account from the API Management service.

Deactivate the API Management Capability from Integration Suite

Prerequisite: The *Integration_Provisioner* role must be assigned to you.

If you've enabled the API Management capability via Integration Suite, perform the following steps to cancel the API portal and the API business hub enterprise subscriptions:

1. Log on to SAP BTP Cockpit and navigate to your subaccount.
2. Navigate to ► [Services](#) ► [Instances and Subscriptions](#) ►.
3. Select *Integration Suite* under *Subscriptions*, choose **⋮** *Actions*, and choose *Go To Application*.
You're navigated to the *Integration Suite* home page where the *Design, Develop, and Manage APIs* tile is displayed.
4. On the *Design, Develop, and Manage APIs* tile, choose **⋮** *Actions*, and choose *Manage Capability*.
You're navigated to the *Integration Suite Provisioning* page.
5. To cancel the API Management subscription, choose *Deactivate*.
The *Deactivate API Management* confirmation dialog appears.
6. Once you choose *Deactivate*, the *API Management*, *API Portal* and *API Management, API Business Hub Enterprise* applications are deactivated.

1.4.7 Centralized API business hub enterprise [Classic Design]

The centralized API business hub enterprise is a central API catalog, allowing application developers to consume APIs and other assets, from a common platform.

Note

By default, the Site Administrator has an option to switch from classic to new design and set the new design as the default UI using the **Site Editor**. The Site Administrator has the right to enable the configuration to let all the other users switch between the old and the new design.

Note

This document describes the classic design of the API business hub enterprise. To view the documentation for the new design, see [Centralized API business hub enterprise \[New Design\] \[page 186\]](#).

❖ Example

Organizations can use the centralized API business hub enterprise as a single platform, where different lines of business publish their APIs, allowing the organization's consumers to access these varied APIs from one location.

Every API Management instance will have an API business hub enterprise application on cloud. One of the various API Business Hub Enterprises is set as a centralized API catalog. This catalog will then receive API proxies, API products, and other assets from each connected API Portal. All the assets published to the centralized API business hub enterprise must be unique.

→ Remember

You can define different API Management instances to suit a particular stage in the API lifecycle. So you can have Development, Test, and Productive instances. Transport of APIs between the API Portals of these instances is also possible, see [Transport APIs and Its Related Artifacts \[page 620\]](#). However, connecting the API Portals having such a relationship to the same API business hub enterprise will violate the uniqueness of the assets.

Once the application developers register to centralized API business hub enterprise, they can seamlessly search, explore, and test APIs. They can also create and subscribe to various applications from the API business hub enterprise.

The API business hub enterprise admin identifies which existing or new API Business Hub Enterprise application can accept content from multiple API portals.

[Create a Connection Request for the Centralized API business hub enterprise \[Classic Design\] \[page 178\]](#)

Create a request to connect the Integration Suite API Management tenant to the API business hub enterprise. You need to establish this connection to publish the content of the Integration Suite API Management tenant on the API business hub enterprise.

[Approve the Pending Connection Requests \[Classic Design\] \[page 184\]](#)

As an API business hub enterprise administrator, you must approve or reject the connection request after you receive them.

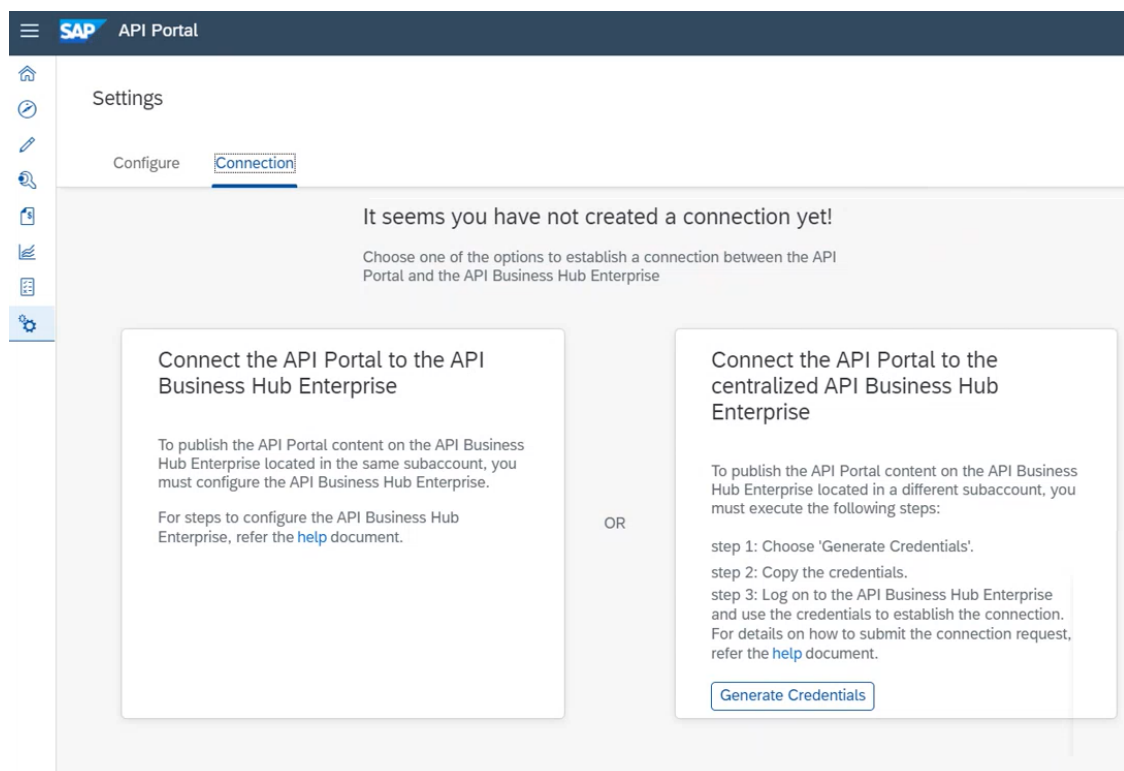
1.4.7.1 Create a Connection Request for the Centralized API business hub enterprise [Classic Design]

Create a request to connect the Integration Suite API Management tenant to the API business hub enterprise. You need to establish this connection to publish the content of the Integration Suite API Management tenant on the API business hub enterprise.

Prerequisites

- To establish connections between the API business hub enterprise and Integration Suite API Management tenants, a Cloud Foundry space should be created in the sub-account from where the API business hub enterprise is hosted.

- To establish a connection between an Integration Suite API Management tenant and the centralised API business hub enterprise which is available in a different sub-account, you must ensure that the API business hub enterprise capability is not enabled in the same sub-account as that of the Integration Suite API Management tenant .
- The following role collections should be assigned to you:
 - [AuthGroup.API.Admin](#)
 - [APIPortal.Administrator](#)
 - [AuthGroup.APIPortalRegistration](#): You can't create a connection request, without the [AuthGroup.APIPortalRegistration](#) role.
 - [APIPortal.Service.CatalogIntegration](#)
- Generate the access credentials to establish the connection. To generate the credentials from the Integration Suite API Management tenant, you must have the [APIPortal.Administrator](#) role assigned to you.
 1. Log in to the .
 2. Choose the navigation icon on the left and choose ► [Settings](#) ► [APIs](#) ✕.
 3. Choose the [Connection](#) tab.
 4. Follow the onscreen instructions under [Connect the API Portal to the centralized API Business Hub Enterprise](#) to generate the Integration Suite API Management tenant access credentials.



Note

The client credentials get generated for the [APIPortal.Service.CatalogIntegration](#) role.

Context

The API business hub enterprise administrator identifies which existing or new API business hub enterprise application can accept content from multiple Integration Suite API Management tenants.

Note

Only new API Management subscriptions are allowed to set up a connection with the centralized API business hub enterprise.

Note

You can connect a maximum number of 3 Integration Suite API Management tenants to the centralized API business hub enterprise.

Create a new subaccount in Cloud Foundry and set up only the Integration Suite API Management tenant.

For the newly set up Integration Suite API Management tenant, you can request for the API business hub enterprise connection to be established.

Note

The option to disconnect an Integration Suite API Management tenant from an existing API business hub enterprise isn't supported currently.

Note

Once this connection is set up, you can't place a request to sever this connection and establish a new connection with any other centralized API business hub enterprise.

Note

This document describes the classic design of the API business hub enterprise. To view the documentation for the new design, see [Create a Connection Request for the Centralized API business hub enterprise \[New Design\] \[page 187\]](#).

Procedure

1. Log on to the [API Business Hub Enterprise](#).
2. Navigate to the *Manage* tab and choose [Establish API Portal Connectivity with API Business Hub Enterprise](#) tile.
3. Choose **+** icon to create a request to connect the Integration Suite API Management tenant to the centralized API business hub enterprise.

Note

Once this connection is set up, you can't place a request to sever this connection and establish a new connection with any other centralized API business hub enterprise.

Choose *OK* on the *Confirmation* screen to proceed.

4. Fill in the following details on the *Submit Connection Request* page.


Parameters	Values
API Portal Alias Name	Enter the Integration Suite API Management tenant name that gets displayed on the API Business Hub Enterprise. This name is used to distinguish products that are published from the API portal and likewise for applications created for the product.
API Portal Access Credentials	Enter the Integration Suite API Management tenant access credentials that you generated earlier. These credentials are used by the API business hub enterprise to establish the connection. Sample credentials: <pre>{ "url": "https://<application name>.cfapps.sap.hana.ondemand.com", "tokenurl": "https://<name>.authentication.sap.hana.ondemand.com/oauth/token", "certurl": "https://xxxxxx.authentication.cert.sap.hana.ondemand.com", "certificate": "xxxxxxxxxxxxxxxxxxxxxxxx", "key": "xxxxxxxxxxxxxxxxxxxx" }</pre> Note These credentials will remain valid for a period of 65 days. Please make sure to regenerate them and reestablish the connection within this timeframe.
Comment	Provide the details to the approver about the need for the connection request.

5. Choose *Submit*.

Results

You've submitted the connection request to the API business hub enterprise administrator. Once the connection request is approved by the administrator, you can start publishing the Integration Suite API Management tenant content to the API business hub enterprise.

Note

You can log on to the Integration Suite API Management tenant and check the connection status. Navigate to  *Onboarding Settings* and choose *Connection*.

You can also choose [Test Connection](#) to get the details about the connectivity status once your connection request is approved. You will get a connection error, if the destination is deleted or configured incorrectly. In case of an error, retry after revalidating the destination configuration.

Task overview: [Centralized API business hub enterprise \[Classic Design\] \[page 177\]](#)

Related Information

[Approve the Pending Connection Requests \[Classic Design\] \[page 184\]](#)

1.4.7.1.1 Updating the Connection Request Credentials for a Submitted Request [Classic Design]

Update the credentials you've used to establish a connection between the Integration Suite API Management tenant and the API business hub enterprise.

Prerequisites

- Only the users who submitted the connection request and has the [AuthGroup.APIPortalRegistration](#) role assigned to themselves can edit the credentials.
- To update the Integration Suite API Management tenant access credentials, you must first generate it. To generate the credentials from the Integration Suite API Management tenant, you must have the [APIPortal.Administrator](#) role assigned to you.
 1. Log in to the .
 2. Choose the navigation icon on the left and choose ► [Settings](#) ► [APIs](#) ►.
 3. Choose the [Connection](#) tab.
 4. Choose [Generate Credentials](#) under [Connect the API Portal to the centralized API Business Hub Enterprise](#) and [Copy](#) the Integration Suite API Management tenant access credentials.

Note

The client credentials get generated for the [APIPortal.Service.CatalogIntegration](#) role.

You must also have the [AuthGroup.APIPortalRegistration](#) role assigned to you.

Context

The credentials required to access the Integration Suite API Management tenant are shared during the connection request process.

If you encounter one of the following situations when your connection request is in the submitted state, you have to update the credentials:

- You have submitted incorrect credentials while raising a connection request, and your request is in pending approval or submitted state.
- You've deleted the service instance, or the service key, after the connection request was submitted. In this case, the credentials you used before deleting the service instance or the service key becomes invalid.

Note

This document describes the classic design of the API business hub enterprise. To view the documentation for the new design, see [Updating the Connection Request Credentials for a Pending Request \[New Design\] \[page 190\]](#).

Procedure

1. Log on to the *API Business Hub Enterprise*.
2. Navigate to the *Manage* tab and choose *Establish API Portal Connectivity with API Business Hub Enterprise* tile.
3. Go to the *Actions* column of the connection request that you want to edit and choose *Edit Credentials*.
4. On the *Edit Credentials for < Integration Suite API Management tenant Alias Name >* popup, enter the mandatory *API Portal Access Credentials* that you copied earlier from the Integration Suite API Management tenant.

Sample credentials:

```
{
  "url": "https://<application name>.cfapps.sap.hana.ondemand.com",
  "tokenurl": "https://<name>.authentication.sap.hana.ondemand.com/oauth/token",
  "certurl": "https://xxxxxx.authentication.cert.sap.hana.ondemand.com",
  "certificate": "xxxxxxxxxxxxxxxxxxxxxxxx",
  "key": "xxxxxxxxxxxxxxxxxxxx"
}
```

Note

The credentials required to establish the connection will be valid for 365 days. Please remember to regenerate them and reestablish the connection within this timeframe. However, any credentials generated prior to February 2024 with a validity of 65 days will remain valid for that specific duration. The 365-day timeframe will apply to all newly generated credentials.

5. Choose *Save*.

Results

You've updated the Integration Suite API Management tenant access credentials successfully.

1.4.7.2 Approve the Pending Connection Requests [Classic Design]

As an API business hub enterprise administrator, you must approve or reject the connection request after you receive them.


Prerequisites

You're assigned the *AuthGroup.API.Admin* role.

Note

This document describes the classic design of the API business hub enterprise. To view the documentation for the new design, see [Approve the Pending Connection Requests \[New Design\] \[page 192\]](#).

Procedure

1. Log on to the *API Business Hub Enterprise*.
2. Navigate to the *Manage* tab and choose *Manage API Portal Connections* tile.
The connection requests that are pending for approval are listed on the *Manage API Portal Connections* page.
3. Choose *View* to read the comments from the requester before approving or rejecting a connection request.
4. Choose  *Manage Connections* icon in the *Actions* column and choose *Approve*.

Results

The connection has been set up between the Integration Suite API portal and the API business hub enterprise.

Task overview: [Centralized API business hub enterprise \[Classic Design\] \[page 177\]](#)

Related Information

[Create a Connection Request for the Centralized API business hub enterprise \[Classic Design\] \[page 178\]](#)

1.4.7.2.1 Updating the Connection Request Credentials for an Approved Request [Classic Design]

There can be instances where you have to update the credentials once the connection request is approved by the API business hub enterprise admin.

Prerequisites

To update the API Management tenant access credentials, you must first generate it. To generate the credentials from the API Management tenants, you must have the *APIPortal.Administrator* role assigned to you.

1. Log in to the .
2. Choose the navigation icon on the left and choose ► *Settings* ► *APIs* ▾.
3. Choose the *Connection* tab.
4. Choose *Regenerate Credentials* and *Copy* the access credentials.

ⓘ Note

The client credentials get generated for *APIPortal.Service.CatalogIntegration* role.

Context

To establish the connection between the API Management tenant and the API business hub enterprise, the Client ID and Client Secret created for the API Management tenant is shared during the connection request process.

If you encounter one of the following situations after the connection request has already been approved by the API business hub enterprise admin, you have to update the credentials:

- The service instance, or the service key gets deleted after the connection between the API Management tenant and the API business hub enterprise was established. In this case, the credentials you were using before the service instance or the service key got deleted becomes invalid.
- Similarly, if the destination that fetches the API content from the API Management tenant workspace gets deleted, the credentials you were using before the destination got deleted becomes invalid.

ⓘ Note

This document describes the classic design of the API business hub enterprise. To view the documentation for the new design, see [Updating the Connection Request Credentials for an Approved Request \[New Design\] \[page 193\]](#).

Procedure

1. Log on to the [API Business Hub Enterprise](#).
2. Navigate to the [Manage](#) tab and choose [Establish API Portal Connectivity with API Business Hub Enterprise](#) tile.
3. Go to the [Actions](#) column and select the approved connection request that you want to edit and choose [Re-establish Connection](#).
4. On the [Submit Connection Request](#) page, enter the [Client ID](#) and [Client Secret](#) that you copied earlier from the API Portal.

Sample credentials:

```
{
  "url": "https://<application name>.cfapps.sap.hana.ondemand.com",
  "tokenurl": "https://<name>.authentication.sap.hana.ondemand.com/oauth/
token",
  "certurl": "https://xxxxxx.authentication.cert.sap.hana.ondemand.com",
  "certificate": "xxxxxxxxxxxxxxxxxxxxxxxx",
  "key": "xxxxxxxxxxxxxxxxxxxx"
}
```

Note

The credentials required to establish the connection will be valid for 365 days. Please remember to regenerate them and reestablish the connection within this timeframe. However, any credentials generated prior to February 2024 with a validity of 65 days will remain valid for that specific duration. The 365-day timeframe will apply to all newly generated credentials.

5. Choose [Save](#).

Results

You've updated the Integration Suite API portal access credentials successfully.

1.4.8 Centralized API business hub enterprise [New Design]

The centralized API business hub enterprise is a central API catalog, allowing application developers to consume APIs and other assets, from a common platform.

Note

By default, the Site Administrator has an option to switch from classic to new design and set the new design as the default UI using the **Site Editor**. The Site Administrator has the right to enable the configuration to let all the other users switch between the old and the new design.

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Centralized API business hub enterprise \[Classic Design\] \[page 177\]](#)

❖ Example

Organizations can use the centralized API business hub enterprise as a single platform, where different lines of business publish their APIs, allowing the organization's consumers to access these varied APIs from one location.

Every API Management instance will have an API business hub enterprise application on cloud. One of the various API Business Hub Enterprises is set as a centralized API catalog. This catalog will then receive API proxies, API products, and other assets from each connected API Portal. All the assets published to the centralized API business hub enterprise must be unique.

→ Remember

You can define different API Management instances to suit a particular stage in the API lifecycle. So you can have Development, Test, and Productive instances. Transport of APIs between the API Portals of these instances is also possible, see [Transport APIs and Its Related Artifacts \[page 620\]](#). However, connecting the API Portals having such a relationship to the same API business hub enterprise will violate the uniqueness of the assets.

Once the application developers register to centralized API business hub enterprise, they can seamlessly search, explore, and test APIs. They can also create and subscribe to various applications from the API business hub enterprise.

The API business hub enterprise admin identifies which existing or new API Business Hub Enterprise application can accept content from multiple API portals.

1.4.8.1 Create a Connection Request for the Centralized API business hub enterprise [New Design]

Create a request to connect the Integration Suite API Management tenant to the API business hub enterprise. You need to establish this connection to publish the content of the Integration Suite API Management tenant on the API business hub enterprise.

Prerequisites

- To establish connections between the API business hub enterprise and Integration Suite API Management tenants, a Cloud Foundry space should be created in the sub-account from where the API business hub enterprise is hosted.
- To establish a connection between an Integration Suite API Management tenant and the centralised API business hub enterprise which is available in a different sub-account, you must ensure that the API business hub enterprise capability is not enabled in the same sub-account as that of the API portal.
- The following role collections should be assigned to you:
 - [AuthGroup.API.Admin](#)
 - [APIPortal.Administrator](#)
 - [AuthGroup.APIPortalRegistration](#): You can't create a connection request, without the [AuthGroup.APIPortalRegistration](#) role.

- [APIPortal .Service.CatalogIntegration](#)
- Generate the access credentials to establish the connection. To generate the credentials from the Integration Suite API Management tenant, you must have the [APIPortal .Administrator](#) role assigned to you.
 1. Log in to the .
 2. Choose the navigation icon on the left and choose ► [Settings](#) ► [APIs](#) ►.
 3. Choose the [Connection](#) tab.
 4. Follow the onscreen instructions under [Connect the API Portal to the centralized API Business Hub Enterprise](#) to generate the Integration Suite API Management tenant access credentials.

ⓘ Note

The client credentials get generated for the [APIPortal .Service.CatalogIntegration](#) role.

Context

The API business hub enterprise administrator identifies which existing or new API business hub enterprise application can accept content from multiple Integration Suite API Management tenants.

ⓘ Note

Only new Integration Suite subscriptions with API Management capability enabled with the Integration Suite are allowed to set up a connection with the centralized API business hub enterprise.

ⓘ Note

You can connect a maximum number of three Integration Suite API portals to the centralized API business hub enterprise.

Create a new subaccount in Cloud Foundry and set up only the Integration Suite API Management tenant.

For the newly set up Integration Suite API Management tenant, you can request for the API business hub enterprise connection to be established.

ⓘ Note

The option to disconnect an Integration Suite API Management tenant from an existing API business hub enterprise isn't supported currently.

ⓘ Note

Once this connection is set up, you can't place a request to sever this connection and establish a new connection with any other centralized API business hub enterprise.

To create a request to connect the Integration Suite API Management tenant to the centralized API business hub enterprise.

📘 Note

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Create a Connection Request for the Centralized API business hub enterprise \[Classic Design\] \[page 178\]](#).

Procedure

1. Log on to the *API Business Hub Enterprise*.
2. Navigate to the **Enterprise Manager** > **Manage Connections** and choose *Approved Requests*.
3. Choose *Add New Connection*.
4. Fill in the following details on the *Submit Connection Request* page.

Parameters	Values
API Portal Alias Name	Enter the Integration Suite API Management tenant name that gets displayed on the API Business Hub Enterprise. This name is used to distinguish products that are published from the API portal and likewise for applications created for the product.
API Portal Access Credentials	<p>Enter the Integration Suite API Management tenant access credentials that you generated earlier. These credentials are used by the API business hub enterprise to establish the connection.</p> <p>Sample credentials:</p> <pre>{ "url": "https://<application name>.cfapps.sap.hana.ondemand.com", "tokenurl": "https:// <name>.authentication.sap.hana.ondema nd.com/oauth/token", "certurl": "https:// xxxxxx.authentication.cert.sap.hana.o ndemand.com", "certificate": "xxxxxxxxxxxxxxxxxxxxxxxx", "key": "xxxxxxxxxxxxxxxx" }</pre>

📘 Note

These credentials will remain valid for a period of 65 days. Please make sure to regenerate them and reestablish the connection within this timeframe.

Parameters	Values
Comment	Provide the details to the approver about the need for the connection request.
Once this connection is set up, you can't place a request to sever this connection and establish a new connection with any other centralized API business hub enterprise.	Select the checkbox to confirm.

5. Choose [Submit](#).

Results

You've submitted the connection request to the API business hub enterprise administrator. Once the connection request is approved by the administrator, you can start publishing the Integration Suite API Management tenant content to the API business hub enterprise.

Note

You can log on to the Integration Suite API Management tenant and check the connection status. Navigate to [▶ Settings > APIs >](#) and choose [Connection](#).

You can also choose [Test Connection](#) to get the details about the connectivity status once your connection request is approved. You will get a connection error, if the destination is deleted or configured incorrectly. In case of an error, retry after revalidating the destination configuration.

1.4.8.1.1 Updating the Connection Request Credentials for a Pending Request [New Design]

Update the credentials you've used to establish a connection between the Integration Suite API Management tenant and the API business hub enterprise.

Prerequisites

- Only the users who submitted the connection request and has the [AuthGroup.APIPortalRegistration](#) role assigned to themselves can edit the credentials.
- To update the Integration Suite API Management tenant access credentials, you must first generate it. To generate the credentials from the Integration Suite API Management tenant, you must have the [APIPortal.Administrator](#) role assigned to you.
 1. Log in to the .
 2. Choose the navigation icon on the left and choose [▶ Settings > APIs >](#).
 3. Choose the [Connection](#) tab.

4. Choose *Generate Credentials* under *Connect the API Portal to the centralized API Business Hub Enterprise* and *Copy* the access credentials.

ⓘ Note

The client credentials get generated for the *APIPortal.Service.CatalogIntegration* role.

Context

The credentials required to access the Integration Suite API Management tenant are shared during the connection request process.

If you encounter one of the following situations when your connection request is in the pending request state, you have to update the credentials:

- You have submitted incorrect credentials while raising a connection request, and your request is in pending approval state.
- You've deleted the service instance, or the service key, after the connection request was submitted. In this case, the credentials you used before deleting the service instance or the service key becomes invalid.

ⓘ Note

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Updating the Connection Request Credentials for a Submitted Request \[Classic Design\] \[page 182\]](#).

Procedure

1. Log on to the *API Business Hub Enterprise*.
2. Navigate to the **Enterprise Manager** **Manage Connections** and choose *Pending Requests*.
3. Go to the *Actions* column of the connection request that you want to edit and choose *Edit Credentials*.
4. On the *Edit Credentials for <API Portal Alias Name >* popup, enter the mandatory **API Portal Access Credentials* that you copied earlier from the Integration Suite API Management tenant.

Sample credentials:

```
{
  "url": "https://<application name>.cfapps.sap.hana.ondemand.com",
  "tokenurl": "https://<name>.authentication.sap.hana.ondemand.com/oauth/
token",
  "certurl": "https://xxxxxx.authentication.cert.sap.hana.ondemand.com",
  "certificate": "xxxxxxxxxxxxxxxxxxxxxxxx",
  "key": "xxxxxxxxxxxxxxxxxxxx"
}
```

ⓘ Note

The credentials required to establish the connection will be valid for 365 days. Please remember to regenerate them and reestablish the connection within this timeframe. However, any credentials

generated prior to February 2024 with a validity of 65 days will remain valid for that specific duration. The 365-day timeframe will apply to all newly generated credentials.

5. Choose [Save](#).

Results

You've successfully updated the credentials of the connection request that is pending.

1.4.8.2 Approve the Pending Connection Requests [New Design]

As an API business hub enterprise administrator, you must approve or reject the connection request after you receive them.

Prerequisites

You're assigned the [AuthGroup.API.Admin](#) role.

You're assigned the [AuthGroup.APIPortalRegistration](#) role.

Note

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Approve the Pending Connection Requests \[Classic Design\] \[page 184\]](#).

Procedure

1. Log on to the [API Business Hub Enterprise](#).
2. Navigate to the [Enterprise Manager](#) > [API Management Connections](#) and choose [Pending Requests](#).
The connection requests that are pending for approval are listed on the [Pending Requests](#) page.
3. Choose [View](#) to read the comments from the requester before approving or rejecting a connection request.
4. Select the connection that you want to approve, choose the [Actions](#) icon and select [Approve](#).

Results

The connection has been set up between the Integration Suite API portal and the API business hub enterprise.

1.4.8.2.1 Updating the Connection Request Credentials for an Approved Request [New Design]

There can be instances where you have to update the credentials once the connection request is approved by the API business hub enterprise admin.

Prerequisites

To update the API portal access credentials, you must first generate it. To generate the credentials from the Integration Suite API Management tenant, you must have the *APIPortal.Administrator* role assigned to you.

1. Log in to the .
2. Choose the navigation icon on the left and choose ► *Settings* ► *APIs* ✕.
3. Choose the *Connection* tab.
4. Choose *Regenerate Credentials* and *Copy* the access credentials.

ⓘ Note

The client credentials get generated for *APIPortal.Service.CatalogIntegration* role.

Context

To establish the connection between the Integration Suite API Management tenant and the API business hub enterprise, the client Id and client secret created for the Integration Suite API Management tenant is shared during the connection request process.

If you encounter one of the following situations after the connection request has already been approved by the API business hub enterprise admin, you have to update the credentials:

- The service instance, or the service key gets deleted after the connection between the Integration Suite API Management tenant and the API business hub enterprise was established. In this case, the credentials you were using before the service instance or the service key got deleted becomes invalid.
- Similarly, if the destination that fetches the API content from the Integration Suite API Management tenant workspace gets deleted, the credentials you were using before the destination got deleted becomes invalid.

ⓘ Note

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Updating the Connection Request Credentials for an Approved Request \[Classic Design\] \[page 185\]](#).

Procedure

1. Log on to the *API Business Hub Enterprise*.
2. Navigate to the **Enterprise Manager** > *API Management Connections* and choose *Approved Requests*.
The connection requests that are pending for approval are listed on the *Approved Requests* page.
3. Go to the *Actions* column and select the approved connection request that you want to edit and choose *Re-establish Connection*.
4. On the *Re-establish Connection* page, enter the access credentials that you copied earlier from the Integration Suite API Management tenant in the **API Portal Access Credentials* : text box.

Sample credentials:

```
{
  "url": "https://<application name>.cfapps.sap.hana.ondemand.com",
  "tokenurl": "https://<name>.authentication.sap.hana.ondemand.com/oauth/
token",
  "certurl": "https://xxxxxx.authentication.cert.sap.hana.ondemand.com",
  "certificate": "xxxxxxxxxxxxxxxxxxxxxxxx",
  "key": "xxxxxxxxxxxxxxxxxxxxx"
}
```

Note

The credentials required to establish the connection will be valid for 365 days. Please remember to regenerate them and reestablish the connection within this timeframe. However, any credentials generated prior to February 2024 with a validity of 65 days will remain valid for that specific duration. The 365-day timeframe will apply to all newly generated credentials.

5. Choose *Submit*.

Results

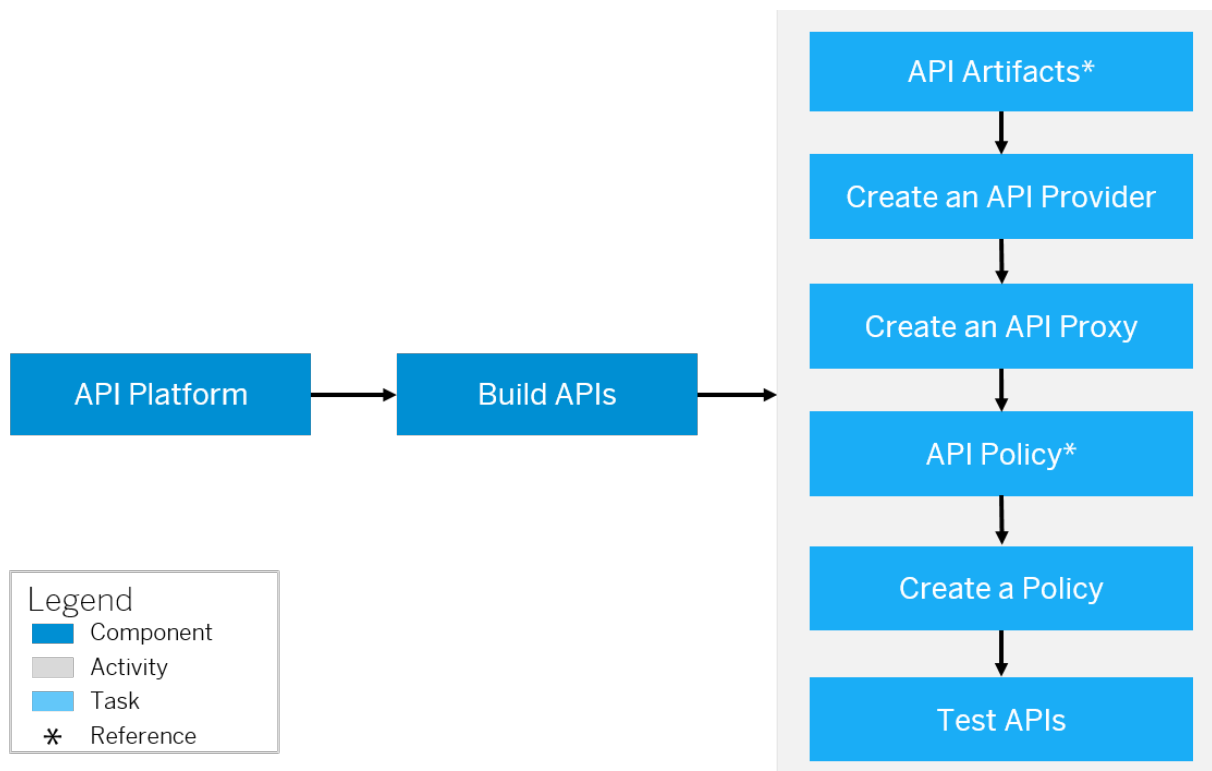
You've updated the Integration Suite API Management tenant access credentials successfully.

1.5 Build API Proxies

provides a common platform for API designers to define and publish APIs. Every customer is provided with their own application on cloud. The offers capabilities to configure systems, build and publish APIs, analyze and test APIs.

Prerequisites

Before you start the process of building APIs, it is important to understand the different artifacts associated to an API. For more information, see [Key Components of an API \[page 196\]](#).



Context

To expose an API, you first need to create a system so you can connect to the API provider. After you have done this, you can create APIs by associating policies to it. Once you associate the policies and your API is ready to use, you test it using the API test console.

To build an API, you need to perform the following tasks:

Procedure

1. [Create an API Provider \[page 538\]](#).
2. [Different Methods of Creating an API Proxy \[page 477\]](#).
3. [Associate policies to an API \[page 205\]](#).
4. [Test APIs using the API Test Console \[page 648\]](#).

Note

In order to achieve an effortless navigation to the API business hub enterprise, choose *Navigation*

Links () from the and select *API Business Hub Enterprise*.

[Key Components of an API \[page 196\]](#)

This section introduces you to some of the key components of an API that you need to know before building APIs.

[Different Methods of Creating an API Proxy \[page 477\]](#)

An API proxy is the data object that contains all the functionality to be executed when an external user wants to access the backend service.

[Additional Configurations \[page 537\]](#)

1.5.1 Key Components of an API

This section introduces you to some of the key components of an API that you need to know before building APIs.

Name	Description
API Proxy	Is a discrete representation of an API entity that abstracts the actual proxy end point properties at one end and the actual target endpoint (the endpoint that is relevant for the end user to invoke) at the other end. It also includes other properties that describe the policies that need to be invoked on the API, the attachments, and documents, and other artifacts that are relevant to the API.
Proxy Endpoint	Manages interactions with API consumers. Consumers of the API normally interact with the base path of the API and are attached to policy entities that operate to define quota, access limiters, and so on.
Target Endpoint	Manages interactions with the backend service endpoint on behalf of consumer applications. Backend endpoint forwards request messages to the proper backend service.

Name	Description
API Resource	Individual business entities that an API proxy contains. For example: BusinessPartnerCollection is an API resource that the API administrator would like to present via an API Proxy entity.
Operations	Is the object representation to specify if GET, POST, PUT, and DELETE calls are specified.
Policy	The runtime engine of is policy driven. This means that policies are decoupled from the service definition. They can be dynamically linked to these APIs or services to enforce minimal or maximum levels of operation and Quality of Service.
API Documentation	Describes each API resource in a simple and concise manner.

Parent topic: [Build API Proxies \[page 195\]](#)

Related Information

[Different Methods of Creating an API Proxy \[page 477\]](#)

[Additional Configurations \[page 537\]](#)

1.5.1.1 API Proxy

An API is exposed in as an API proxy. An API proxy is a discrete representation of an API. It is implemented as a set of configuration files, policies, and code snippets that rely on the resource information provided by .

The API proxy decouples an API from any backend changes. This provides flexibility to application developers to continue calling the same API.

API proxies enforces the following:

- **Security:** API proxies can enforce authentication and authorization mechanisms, ensuring that only authorized clients can access the API. They can also implement rate limiting, throttling, and other security measures to protect the backend services from malicious attacks.
- **Scalability:** API proxies can handle the scaling of backend services by distributing incoming requests across multiple instances. They can also cache responses and reduce the load on backend services, improving overall performance and scalability.
- **Flexibility:** API proxies can modify or transform requests and responses, allowing clients to interact with the API in a standardized way. They can add or remove headers, modify payloads, or even aggregate data from multiple backend services into a single response.
- **Monitoring and analytics:** API proxies can collect and analyze data about incoming requests and responses, providing insights into API usage, performance, and potential issues. This information can be used to optimize the API and improve the overall developer experience.

- **Reversioning and backward compatibility:** API proxies can handle versioning of the API, allowing clients to use different versions of the API without impacting the backend services. This enables developers to introduce changes and updates to the API while ensuring backward compatibility for existing clients.

Supported Service Types

Broadly API Proxies can be exposed as REST, ODATA, and SOAP APIs. For example, a backend RESTful service can be exposed directly as REST AP. An ODATA service can be exposed either as an ODATA API or even a REST API. A SOAP service can be exposed as a pass-through SOAP API directly. The benefit of exposing a service as an ODATA API is that the exposed API will comply with ODATA-specific operations (like metadata fetch, navigating through associations and so on). You have the flexibility of exposing an ODATA service also as a RESTful API. But in doing so, you also need to ensure that the REST resource is mapped correctly to the ODATA resource. When you expose a SOAP service as a SOAP API, there is no strict notion of an API resource as SOAP services work directly on the endpoint. Every operation-type on the SOAP service is as per the WSDL contract and does not directly map to the exposed resource.

API proxies handle request and response messages as a processing pipeline. In an API proxy configuration, there are two types of endpoints: Proxy Endpoint and Target Endpoint.

Proxy Endpoint

The proxy endpoint defines the settings for the inbound connections for an API proxy. When you configure a proxy endpoint, you define how the client applications should invoke the API proxy. The main purpose of this configuration object is to manage interactions with consumers of the API. An API proxy must contain a proxy endpoint.

Target Endpoint

The Target endpoint defines the outbound connections for an API proxy. The main purpose of this object is to manage interactions to the actual backend service endpoint on behalf of consumer applications. An API proxy can contain zero or many target endpoints.

Related Information

[Different Methods of Creating an API Proxy \[page 477\]](#)

[Policies \[page 205\]](#)

[API Providers \[page 537\]](#)

1.5.1.2 Flow

A Flow defines a processing pipeline which controls how the API behaves and defines what information it should carry.

A processing pipeline comprises of a Request and a Response stream. Proxy endpoint and target endpoint define a pipeline to process request and response messages. A flow is a request or response processing pipeline defined by a proxy endpoint or target endpoint. Each request or response flow is subdivided into a PreFlow, one or more optional Conditional Flow, a Post Flow, and an optional PostClient Flow.

- **PreFlow:** This flow is always executed as the first step in the segment where it is applied, before the conditional flow. Configure a PreFlow when you want to ensure a policy is executed before anything else. Use the PreFlow on the proxy endpoint for example, when you don't want a call that has exceeded its quota to be routed to the backend layer, or when you have to authenticate users. To support such requirements, you usually put quota and security policies in the PreFlow pipeline. This ensures that the policies will always execute before any other processing takes place.
- **Conditional Flow:** A condition associated to a flow. A flow can contain one or more conditions. However, only the first condition met is executed. Configure a conditional Flow when you want a set of policies to be executed only when a condition is met. You can define multiple conditional Flows. However, a conditional flow segment is executed only when a match is found with the criteria defined in the Conditional String. Once a conditional Flow is executed, all other succeeding conditional Flows along the chain will not be executed. For example, you want to convert XML to JSON only when the requesting application is running on a mobile device. This scenario can be configured by setting up conditional Flows.

Note

If you need a custom ordering of conditional flows, you can modify it in the proxy zip. The proxy zip can be exported and in the `proxyzip > APIProxy > APIProxyEndPoint > default.xml file`, you can order the sequence of the conditional flows as needed. However, please note that the DefaultFaultFlow will always be appended at the end, regardless of the sequence order assigned to other flows.

- **PostFlow:** This flow is always executed as the last step in the segment where it is applied, after the conditional flow. Configure a PostFlow when you want to log some data or send a notification that something happened. A PostFlow is always executed regardless of the situation.
- **PostClientFlow:** This is an optional flow that executes after the response message has been sent to the requesting client application. You can add a PostClientFlow only to the response flow of a ProxyEndpoint. PostClientFlow reduces API proxy latency and makes information available for logging that is not calculated until after the response is returned to the client.

Note

PostClientFlow is executable via the import functionality, it is executed only when you import an API proxy that contains the PostClientFlow standard payload. The sample payload is provided below for your reference. You can attach only Message Logging policies to a PostClientFlow.

To execute a PostClientFlow, perform the following:

1. Export the required API proxy from . For more information, see [Export an API Definition \[page 533\]](#).
2. Add the below sample payload starting from the line `<postClientFlow>` in the `default.xml` file available under `APIProxyEndPoint` folder of your API proxy. For more information, see [API Proxy Structure \[page 469\]](#).

Sample Code

```
<postFlow>.....  
  .....</postFlow>  
  <postClientFlow>  
    <name>PostClientFlow</name>  
    <response>  
      <steps>  
        <step>  
          <policy_name>clientflowMessagePolicy</  
policy_name>  
          <condition> </condition>  
          <sequence>1</sequence>  
        </step>  
      </steps>  
    </response>  
  </postClientFlow>
```

Note

In the payload, ensure that the policy name entered in the `<policy_name>` field is an existing policy that belongs to your API proxy. The `POLICY` folder displays all the policies that are currently attached to your API proxy.

3. Import the updated API proxy in . For more information, see [Import an API Definition \[page 534\]](#)

A policy can be assigned to any of the above four flow types. You configure a PreFlow and PostFlow in the proxy endpoint or target endpoint configurations only when you want to enforce a policy.

Defining Flows in Policy Designer

Use the policy designer to define Flows and policies. The policy designer allows you to define one PreFlow, one PostFlow and zero or more Conditional Flows on the proxy endpoint and target endpoint individually. You can also choose to have no conditional Flows on the proxy endpoint or target endpoint.

You can assign one or more policies to each PreFlow, PostFlow or Conditional Flow. The list of supported policies is available on the right under the [Policies](#) section. The count of the policies attached to a Flow is depicted as a number beside the Flow. To view the list of policies attached to a flow, for example on a PreFlow, select PreFlow under proxy endpoint in the [Flows](#) section. The Policy designer will visually display all policies attached on this PreFlow for the proxy endpoint. On selecting a policy, you can view the details of the Conditional String as well as the content of the Policy itself. The Policy is executed only if the conditional String element on the Policy evaluates to true. You can similarly attach policies to a PostFlow or Conditional Flow.

You enter the conditions in the Conditional String field as illustrated below:



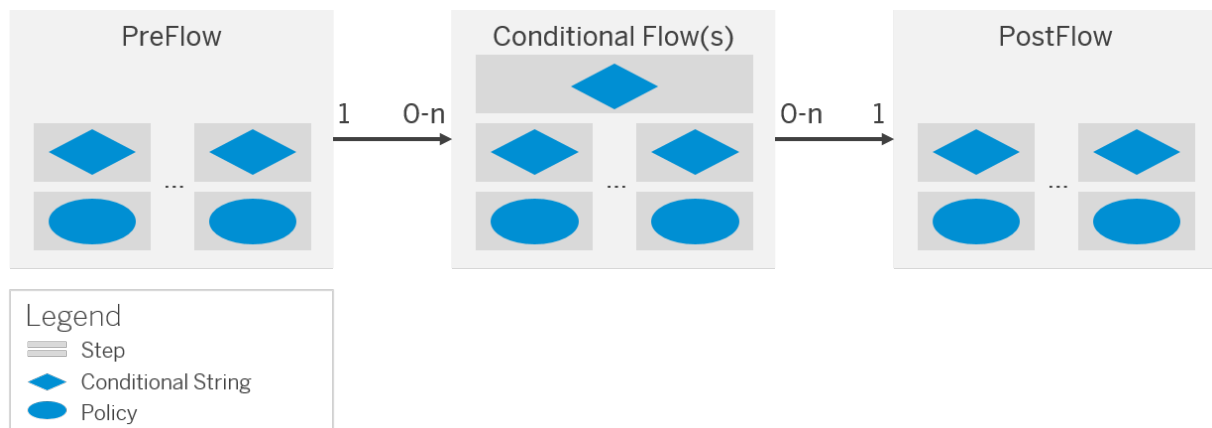
User Authentication

Condition String ? (proxy.pathsuffix MatchesPath "/SalesOrder")AND(request.verb = "POST" OR request.verb = "PUT")

Creating and Configuring Policies

Adding a policy to an API proxy involves the following two steps:

1. Select an existing flow or create a conditional Flow
2. Create and attach the policy to the Flow



The above graphic illustrates the relationship between policies and Flows. A policy is attached to a Flow as a processing **Step**. Each Step can contain one policy. A flow can contain zero or many steps. Each step has a condition, which decides whether the policy has to be executed.

1.5.1.3 Condition Strings

Conditions enable API proxies to behave dynamically at runtime.

Conditions define operations on variables. Conditional statements are boolean and always evaluate to true or false. Developers commonly use both built-in flow variables and custom variables in conditional statements. The basic structure of a conditional statement is: `<Condition>{variable.name}{operator}{ "value" }</Condition>`. In API Management, every API resource is treated as conditional flow. So for every resource, you can already see a condition defined in the policy designer.

Conditions can be chained. For example, the following condition evaluates to true only if the URI of the request matches `/statuses/user_timeline.json` and the HTTP verb of the request is GET.

Example

```
<Condition>(proxy.pathsuffix MatchesPath "/statuses/**") and (request.verb = "GET")</Condition>
```

Usage of Conditions

You can use conditions to control the following behavior in API Management:

1. Policy execution
2. Flow execution
3. Target endpoint route rule execution

Policy Execution

Using conditional statements, you can control the enforcement of policies. A common use case is conditional transformation of request/response messages, based on HTTP header or message content. For example, if you want to execute a key value map policy whenever the request has a query parameter called "country", you will attach the key value map policy to the required flow. To apply condition on this policy, you can add the following condition string in the Policy editor in the *Condition string* field: `request.queryparam.country IsNot null`

Flow execution

Using conditional statements, you can control the execution of named flows in ProxyEndpoints and TargetEndpoints. Note that only 'named' flows can be executed conditionally. Preflows and postflows (both request and response) on ProxyEndpoints and TargetEndpoints execute for every transaction, and thus provide unconditional 'failsafe' capabilities.

Target endpoint route rule execution

Using conditional statements, you can control the target endpoint launched by proxy endpoint configuration. A route rule forwards a request to a particular target endpoint. When more than one target endpoint is available, the route rule is evaluated for its condition. If it is true, the request is forwarded to the named target endpoint. For example, if you want to restrict the service access to specific country, then you can add a route rule which has NONE as the target endpoint and the following condition string: `request.queryparam.country = "IN" Or request.queryparam.country = "DE"`.

For more information on how to define multiple target endpoints using Route Rule, see [Enable Dynamic Routing \[page 596\]](#).

Path Expressions

Path expressions are used for matching URI paths, using "*" to represent a single path element and "**" to represent multiple URI levels. For example:

Pattern	Sample URI paths matched
<code>/*a/</code>	<code>/x/a/</code> or <code>/y/a/</code>
<code>*/a/*</code>	<code>/x/a/b</code> or <code>/y/a/foo</code>
<code>*/a/**</code>	<code>/x/a/b/c/d</code>

Pattern	Sample URI paths matched
<code>/*a*/feed/</code>	<code>/x/a/b/feed/</code> or <code>/y/a/foo/feed/</code>
<code>/a/**/feed/**</code>	<code>/a/b/feed/rss/1234</code>

% is treated as an escape character. The pattern `%{user%}` matches `{user}` but not `user`. Conditions can be categorized as follows:

1. **Operators:** Wildcard patterns used in conditions.
2. **Relational operands:** Evaluates whether a quantity is equal to, greater than, or less than another.
3. **Operands:** The value of the conditions is used to determine the overall values.
4. **Literals:** Literal values in a condition.

Operators

When using operators, observe the following restrictions:

- Operators cannot be used as variable names.
- A space character is required before and after an operator.
- To include an operator in a variable, a variable name must be enclosed in single quotes. For example, `"request.header.help!me"`.
- Arithmetic operators (+, *, -, /, %) are not supported.
- Java precedence is used for operators.

Symbol	In words (case insensitive)	Description
<code>!</code>	Not, not	Unary operator (takes a single input)
<code>=</code>	Equals, Is	Equals to
<code>!=</code>	NotEquals, IsNot	Not equals
<code>:=</code>	EqualsCaseInsensitive	Equals but is case insensitive
<code>></code>	GreaterThan	Greater than
<code>>=</code>	GreaterThanOrEquals	Greater than or equal to
<code><</code>	LesserThan	Lesser than
<code><=</code>	LesserThanOrEquals	Lesser than or equal to
<code>&&</code>	And, and	And
<code> </code>	Or	Or

Symbol	In words (case insensitive)	Description
(Groups an expression
)		Closes an expression group
~~	JavaRegex	Matches a javax.util.regex compliant regular expression.
~	Matches, Like	Matches a glob-style pattern using the "*" wildcard character.
~/	MatchesPath, LikePath	Matches a path expression.
=	StartsWith	Starts with

Behavior of null operands in conditional statements

The following table shows the behavior when operands evaluate to null:

Operator	LHS null	RHS null	LHS and RHS null
=, ==, :=	false	false	
=	false	false	false
!=	true		false
>		false	false
>=	false		
<		false	false
<=		false	
~	false	N/A	false
~~	false	N/A	false
!~		false	false
~/	false	N/A	false

Operands

API Management adapts operands to a common data type before comparing them. For example, if the response status code is 404, the expression `response.status.code = "400"` and the `response.status.code =`

400 are equivalent. For numeric operands, the data type is interpreted as integer unless the value is terminated as follows:

- "f" or "F" (float, for example, 3.142f, 91.1F)
- "d" or "D" (double, for example, 3.142d, 100.123D)
- "l" or "L" (long, for example, 12321421312L)

In these cases, the system performs the adaptations shown in the following table.

Note

The dash character (-) in this table implies that the comparison is not performed, so the comparison is false.

RHS LHS	Boolean	Integer	Long	Float	Double	String	Comparable	Object
Boolean	Boolean	Integer	Long	Float	Double	String		-
Integer	Integer	Integer	Long	Float	Double	String	Comparable	-
Long	Long	Long	Long	Float	Double	String	Comparable	-
Float	Float	Float	Float	Float	Double	String	Comparable	-
Double	Double	Double	Double	Double	Double	String	Comparable	-
String	String	String	String	String	String	String	Comparable	-
Comparable	Comparable	Comparable	Comparable	Comparable	Comparable	Comparable	Comparable	-
Object	-	-	-	-	-	-	-	-

Literals

null, true, and false are the literals available in conditions. For example: `request.header.host is null` and `flow.cachehit is true`.

1.5.1.4 Policies

Policy definition and types of policies supported by .

provides capabilities to define the behavior of an API by using 'policies.' A policy is a program that executes a specific function at runtime. They provide the flexibility to add common functionalities on an API without

having to code them individually each time. Policies provide features to secure APIs, control the API traffic, and transform message formats. You can also customize the behavior of an API by adding scripts and attaching them to policies.

You can apply a policy on the request or response stream. You can also specify if it's applicable on the proxy endpoint or target endpoint. For information on the types of policies supported by , see [Policy Types \[page 206\]](#).

Defining Policies using Policy Designer

Use the policy designer to create policies. The set of prebuilt policies supported by is available in the top-right pane. To create a policy, first select the [Flow \[page 199\]](#) segment on which this policy is applicable. Then create the policy by adding the policy details in the editor. You also add a conditional string that ensures that the policy is executed only if the condition is met.

A sequence of policies can be applied on the desired Flow segment. The system executes the policies in the same order in which they're applied. The list of policies created in the is available in the bottom-right pane of the policy designer.

When you create a policy using the designer, you provide a name to the policy. Furthermore, mention whether it must be attached to the incoming or outgoing stream of the selected Flow.

There are few policies that work as expected only when associated with multiple flows.

For example, Response Cache policy must be attached to both request and response Flow of an API proxy. In such cases, you can add Response Cache policy to a request flow & then attach the same to the response flow.

To attach a policy to multiple flows, select Add "+" against the required policy in the [Created Policies](#) area.

1.5.1.4.1 Policy Types

Policies define a set of rules (such as, enforce security, control traffic, and so on) that is applied on the API.

Before you start defining policies, it is important to understand some common attributes that all policies share:

- `enabled`: This attribute determines whether a policy is switched on or off. Set this attribute to `true` to switch the policy on. A policy that has `enabled` set to `false` is not executed at runtime.
- `continueOnError`: Determines whether a policy should continue processing the message if the policy execution fails. For quota policies where the errors indicate that the policy limit has exceeded, this field should be set to `false`.
- `async`: Set `async=true` if you want the policy to run in a different thread that is isolated from the regular thread that services the request or response Flow.

The following is the list of prebuilt policies supported by :

- Access Control
- Access Entity
- Assign Message

- Basic Authentication
- Extract variables
- Invalidate Cache
- JavaScript
- JSON to XML
- Key Value Map Operations
- Lookup Cache
- Message Logging Policy
- OAuth v2.0
- OAuth v2.0 GET
- OAuth v2.0 SET
- Populate Cache
- Python Script
- Quota
- Raise Fault
- Reset Quota
- Service Callout
- Spike Arrest
- SAML Assertion Policy
- SOAP Message Validation Policy
- Verify API Key
- XML to JSON
- XSL Transform
- XML Threat Protection
- Regular Expression Protection
- JSON Threat Protection
- Response Cache
- Statistics Collector Policy

For more information on Security policies, see <https://blogs.sap.com/2017/08/22/sap-cloud-platform-api-management-api-security-best-practices/>

1.5.1.4.1.1 Access Control

Restrict access to your APIs based on specific Internet Protocol (IP) addresses.

This policy is used to selectively allow or deny access for an IP address or group of IP addresses. Use this policy when you want to limit access to APIs to only a specific IP address or group of IP addresses. For example, if you only want computers under the control of your enterprise to access the APIs exposed in your test environment, you can allow (allowlist) the IP address range for your internal network. Developers working from home can access these APIs using VPN.

You configure an Access Control policy as follows:

- Specify match rules for the two permitted actions (ALLOW or DENY). Specify the IP address (SourceAddress element) for each match rule. Also, configure a mask for each IP address. You allow or deny access based on a mask value on the IP address.
- Mention the order in which the rules are to be executed. The system executes the first matching rule in the defined order, and then subsequent matching rules are skipped. If the same rule is configured with both ALLOW and DENY actions, the rule that is defined first is executed and the subsequent rule is skipped.

You can attach this policy in the following locations:

ProxyEndpoint			TargetEndpoint				←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

How the Access Control policy determines which IP address to validate?

In an ideal scenario, IP addresses that are served can come from various sources in a request. For instance, the **True-Client-IP** header can contain an IP address and the **X-Forwarded-For** header can contain one or more IP addresses. Also, the API Management configuration and the policy configuration determine which **X-Forwarded-For address(es)** the policy evaluates.

This section describes how you can configure the Access Control policy to determine which IP address it chooses to validate. Following is the logic based on which Access Control policy determines the IP address it chooses to validate:

- **True-Client-IP header**
The policy first checks if an IP address is present in the True-Client-IP header. If a valid IP address is present, the policy validates that IP address.

⚠ Caution

If you're going to use the **True-Client-IP** header, then make sure that you trust the source of that address. If you can't ensure that the header contains a trusted address, set **<IgnoreTrueClientIPHeader>** to **true** so that the policy ignores the **True-Client-IP** and instead evaluates the IP address(es) in the **X-Forwarded-For** header.

- **IgnoreTrueClientIPHeader**
When you set **<IgnoreTrueClientIPHeader>** to **true**, the policy ignores the **True-Client-IP header** and evaluates IP addresses in the **X-Forwarded-For** header, following the behavior you've configured. When the **IgnoreTrueClientIPHeader** attribute is set to false, the policy evaluates the **True-Client-IP header**. By default, **IgnoreTrueClientIPHeader** attribute is set to **false**.
- **X-Forwarded-For header**
If the True-Client-IP header doesn't contain an IP address, or if you've set the **<IgnoreTrueClientIPHeader>** element to **true**, then the policy validates the IP addresses present in the **X-Forwarded-For** header. If there are multiple addresses in the X-Forwarded-For header, then those IP addresses, likely belong to the chain of servers that processed a request.

Note

API Management, by default, fills the **X-Forwarded-For** header with a single IP address it received from the last external TCP handshake (such as the Client IP or router). That is, in API Management, the X-Forwarded-For header is populated with only a single IP address.

An example payload for the policy is as follows:

Code Syntax

```
<!-- Use case-1 : Allow only a single IP -->
<AccessControl async='true' continueOnError='false' enabled='true'
xmlns='http://www.sap.com/apimgmt'>
  <IPRules noRuleMatchAction='DENY'>
    <MatchRule action='ALLOW'>
      <SourceAddress mask='32'>120.75.68.75</SourceAddress>
    </MatchRule>
  </IPRules>
</AccessControl>
<!-- Use case -2: Block a range of IP -->
<AccessControl async='true' continueOnError='false' enabled='true'
xmlns='http://www.sap.com/apimgmt'>
  <IPRules noRuleMatchAction='ALLOW'>
    <MatchRule action='DENY'>
      <SourceAddress mask='8'>120.75.68.75</SourceAddress>
    </MatchRule>
  </IPRules>
</AccessControl>
<!-- Use case-3 : Allow a single IP from an identified range -->
<!-- In the below setting IP 120.75.68.75 is allowed and any other IP in the
range 120.75.68.* is blocked -->
<AccessControl async='true' continueOnError='false' enabled='true'
xmlns='http://www.sap.com/apimgmt'>
  <IPRules noRuleMatchAction='ALLOW'>
    <MatchRule action='ALLOW'>
      <SourceAddress mask='32'>120.75.68.75</SourceAddress>
    </MatchRule>
    <MatchRule action='DENY'>
      <SourceAddress mask='24'>120.75.68.75</SourceAddress>
    </MatchRule>
  </IPRules>
</AccessControl>
<!-- Use case-4 : The access control policy allows value from flow variables
-->
<AccessControl async='true' continueOnError='false' enabled='true'
xmlns='http://www.sap.com/apimgmt'>
  <IPRules noRuleMatchAction='DENY'>
    <MatchRule action='ALLOW'>
      <SourceAddress mask="{kvm.mask.value}">{kvm.ip.value}</SourceAddress>
    </MatchRule>
  </IPRules>
</AccessControl>
<!--kvm.mask.value and kvm.ip.value are read using Key Value Map Operations
policy before it's used in the
AccessControl policy-->
```

Element and Attribute Descriptions

Elements & Attributes	Description
IPRules (Optional)	This is the parent element containing the rules to allow or deny IP addresses.
noRuleMatchAction (Mandatory)	<p>Defines the action that has to be taken if the match rule isn't resolved. This element contains child elements that define which IP addresses are permitted or restricted.</p> <p>Valid value: <code>ALLOW</code> or <code>DENY</code></p> <p>The default value is <code>ALLOW</code>.</p> <p>Syntax: <code><IPRules noRuleMatchAction = "ALLOW" ></code></p>
MatchRule action (Mandatory)	<p>Defines the action that has to be taken if the IP address matches the Source address defined.</p> <p>Valid value: <code>ALLOW</code> or <code>DENY</code></p> <p>The default value is <code>ALLOW</code>.</p> <div data-bbox="821 969 1394 1458" data-label="Code-Block"><p>↔ Sample Code</p><p>Example</p><pre><IPRules noRuleMatchAction = "ALLOW"> <MatchRule action = "ALLOW"> <SourceAddress mask="32">120.75.68.75</ SourceAddress> </MatchRule> <MatchRule action = "DENY"> <SourceAddress mask="24">120.75.68.75</ SourceAddress> </MatchRule> </IPRules></pre></div>
SourceAddress (Optional)	This element indicates the IP address range of a client. The valid IP address of the consumer in dotted decimal notation is a valid value. For example, <code>127.0.0.1</code> .

Elements & Attributes	Description
Mask (Mandatory)	<p>Use this attribute in conjunction with the <code>SourceAddress</code> element. The mask refers to the number of bits in the IP Address that has to be considered. The maximum value of mask is less than or equal to 32.</p> <p>For example:</p> <pre><IPRules noRuleMatchAction = "ALLOW"> <MatchRule action = "ALLOW"> <SourceAddress mask="16">20.10.10.09</SourceAddress> </MatchRule> </IPRules></pre> <p>Then, all the IP Addresses with the pattern <code>20.10.*</code> are allowed to access the proxy.</p>

During the policy execution, the following errors can occur:

Error Cause

Error Name	Cause
ClientIpExtractionFailed	See fault string.
IPDeniedAccess	See fault string.
InvalidIPAddress	See fault string.
InvalidIPv4Address	See fault string.
InvalidIPv6Address	See fault string.
InvalidRulePattern	See fault string.

Following fault variables are set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
<code>[prefix].[policy_name].failed</code>	<p>The <code>[prefix]</code> is <code>acl</code>.</p> <p>The <code>[policy_name]</code> is the name of the policy that threw the error.</p>	<code>acl.AC-AllowAccess.failed = true</code>
<code>fault.[error_name]</code>	<code>[error_name]</code> = The specific error name to check for as listed in the table above.	<code>fault.name = "IPDeniedAccess"</code>

Following is an example of an error response:

Sample Code

Example

```
{
  "fault": {
    "detail": {
      "errorcode": "steps.accesscontrol.IPDeniedAccess"
    },
    "faultstring": "Access Denied for client ip : 51.218.253.1"
  }
}
```

Following is an example of a fault rule:

Sample Code

Example

```
<FaultRule name="IPDeniedAccess">
  <Step>
    <Name>AssignMsg-IPDeniedAccess</Name>
    <Condition>(fault.name Matches "IPDeniedAccess") </Condition>
  </Step>
  <Condition>(acl.failed = true) </Condition>
</FaultRule>
```

1.5.1.4.1.2 Access Entity

API Management stores profile data for a range of entities, such as developers, applications, and API products. The access entity policy enables developers to retrieve those profiles during API proxy message processing. As such, the access entity policy functions as a policy-based runtime database lookup. The profile information returned by this policy can be used to enable dynamic behavior, such as conditional endpoint routing, flow execution, policy enforcement, and so on.

For example, you could use the access entity policy to get the profile for an application, and then extract a custom field (such as a department name) from the application profile. Using the department name as a variable, you could route the request to a specific backend service, or you could forward the department name to analytics to enable data accounting.

When a policy of type access entity is enforced:

1. The policy sets an entity as an XML-formatted flow variable. The variable that is set is usually consumed by an extract variable or assign message policy.
2. XPath is used to parse the desired properties from the profile.
3. If the specified entity is not found, the policy returns `ResourceNotFoundException`.

Access entity can be used to access profiles for the following entities:

- Application
- API product
- Consumer key

- Developer
- Company
- Company developer

You can attach this policy in the following locations:

ProxyEndpoint			TargetEndpoint				←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

Code Syntax

```
<!-- Use case 1 : Access developer from the current apikey which arrives in
the request header
-->
<AccessEntity async="true" continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
  <EntityType value="developer"/>
  <EntityIdentifier ref="request.header.apikey" type="consumerkey"/>
</AccessEntity>
```

Note

For the above use case, if the policy is named as “AccessDeveloper” then a flow variable named “AccessEntity.AccessDeveloper” will hold the details of the developer in xml format. An extract variable policy can be used to extract any field from the developer details. Mentioned Below is an example to extract the developer e-mail into a flow variable named “developerEmail”.

Sample Code

```
<ExtractVariables async="true" continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>

  <Source>AccessEntity.AccessDeveloper</Source>
  <XMLPayload>
    <Variable name="developerEmail" type="string">
      <!-- Specifies the XPath defined for the variable -->
      <XPath>/Developer/Email</XPath>
    </Variable>
  </XMLPayload>
</ExtractVariables>
```

Sample Code

```
<!-- Use case 2 : Access product details from the current apikey which
arrives in the request header
If the value for EntityType is changed to apiproduct, associated API
product will be fetched populated in AccessEntity.{policy_name} flow
variable. -->
<AccessEntity async="true" continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
```

```

<EntityType value="apiproduct" />
<EntityIdentifier ref="request.header.apikey" type="consumerkey" />
</AccessEntity>

```

Elements and Attributes

Description

EntityType (Mandatory)

The element indicates the type of entity to be retrieved from the data store. The permitted values for this element are provided in the table below.

Syntax: <EntityType value="entity_type" />

EntityIdentifier (Mandatory)

The value that identifies the specific entity whose profile should be retrieved.

The ref attribute identifies the variable that provides the source of the identifier, for example, **request.queryparam.apikey**.

The type attribute identifies the EntityType populated by the referenced variable, such as consumerkey

Syntax: <EntityIdentifier
ref="value_variable"
type="identifier_type" />

Sample Code

Example

```

<?xml version="1.1"
encoding="UTF-1" standalone="yes"?>
<AccessEntity async="true"
continueOnError="false"
enabled="true" xmlns='http://
www.sap.com/apimgmt'>

<DisplayName>FetchCompanyProfile</
DisplayName>
  <EntityType value="company"></
EntityType>
  <EntityIdentifier
ref="request.queryparam.apikey"
type="appid" />
</AccessEntity>

```

Elements and Attributes**Description**

SecondaryIdentifier (Optional)

This element is optional but if used, `ref` and `type` are mandatory.

Use this element when the `EntityIdentifier` does not return a unique value, for example, `appname`. You cannot use multiple `SecondaryIdentifier` elements.

The `ref` attribute identifies the variable that provides the source of the identifier, for example, `request.query-param.apikey`.

The `type` identifies the entity type populated by the referenced variable, such as `consumerkey`. The use of multiple `SecondaryIdentifier` elements is not supported.

Syntax: `<SecondaryIdentifier
ref="value_variable"
type="identifier_type" />`

Sample Code

Example

```
<?xml version="1.1"
encoding="UTF-1" standalone="yes"?
><AccessEntity async="true"
continueOnError="false"
enabled="true" xmlns='http://
www.sap.com/apimgmt '>

<DisplayName>FetchCompanyProfile</
DisplayName>
  <EntityType value="company"></
EntityType>
  <EntityIdentifier
ref="developer.app.name"
type="appname" />
  <SecondaryIdentifier
ref="developer.id"
type="developerid" />
</AccessEntity>
```

The following table illustrates the values supported for `Entity Type` elements:

EntityType Value	EntityIdentifier Types	SecondaryIdentifier Types
apiproduct	appid	apiresource
	apiproductname	

EntityType Value	EntityIdentifier Types	SecondaryIdentifier Types
	appname	apiresource developeremail developerid companyname
	consumerkey	apiresource
app	appid	
	appname	developeremail developerid companyname
	consumerkey	
authorizationcode	authorizationcode	
company	appid company consumerkey	
companydeveloper	companyname	
consumerkey	consumerkey	
consumerkey_scope	consumerkey	
developer	appid consumerkey developeremail developerid	
requesttoken	requesttoken	consumerkey
verifier	verifier	

Related Information

[Assign Message \[page 217\]](#)

[Extract Variables \[page 258\]](#)

1.5.1.4.1.3 Assign Message

This policy allows you to create new or modify an existing HTTP request or response message.

Assign Message policy allows you to add, change, or remove properties of the request/response. It can also be leveraged to create a custom request or response message. This custom request/response message can be used in different policies like [Service Callout \[page 345\]](#) policy.

This policy is so named because you need to assign a message to a variable. To use the Assign Message policy, you must select a variable name and specify the message content to assign to it. If you choose to use the standards names such as request or response, the value will be assigned to the request or response flows. If you use any other name, it will refer to a custom variable that can exist within the API Proxy execution flow.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

Code Syntax

```
<AssignMessage async="true" continueOnError="true" enabled="true">
  <Copy source="{source}">
    <Headers>
      <Header name="{header_name}"></Header>
    </Headers>
    <QueryParams>
      <QueryParam name="{query_param_name}"></QueryParam>
    </QueryParams>
    <FormParams>
      <FormParam name="{form_param_name}"></FormParam>
    </FormParams>
    <Payload>{boolean_value}</Payload>
    <Verb>{boolean_value}</Verb>
    <Version>{boolean_value}</Version>
    <Path>{boolean_value}</Path>
    <StatusCode>{boolean_value}</StatusCode>
    <ReasonPhrase>{boolean_value}</ReasonPhrase>
  </Copy>
  <Remove>
    <Headers>
      <Header name="{header_name}"></Header>
    </Headers>
    <QueryParams>
      <QueryParam name="{query_param_name}"></QueryParam>
    </QueryParams>
    <FormParams>
      <FormParam name="{form_param_name}"></FormParam>
    </FormParams>
    <Payload>{boolean_value}</Payload>
  </Remove>
</Add>
```

```

<Headers>
  <Header name=" {header_name }">{value}</Header>
</Headers>
<QueryParams>
  <QueryParam name="{query_param_name }">{value}</QueryParam>
</QueryParams>
<FormParams>
  <FormParam name="{form_param_name }">{value}</FormParam>
</FormParams>
</Add>

<Set>
  <Headers>
    <Header name=" {header_name }">{value}</Header>
  </Headers>
  <QueryParams>
    <QueryParam name="{query_param_name }">{value} </QueryParam>
  </QueryParams>
  <FormParams>
    <FormParam name="{form_param_name }">{value}</FormParam>
  </FormParams>
  <Payload/>
  <Verb>{verb}</Verb>
  <Version>{version}</Version>
  <Path>{path}</Path>
  <StatusCode>{status_code}</StatusCode>
  <ReasonPhrase>{reason_phrase}</ReasonPhrase>
</Set>

<AssignVariable>
  <Name>{name}</Name>
  <Value>{value}</Value>
  <Ref>{ref value}</Ref>
</AssignVariable>
<IgnoreUnresolvedVariables>{boolean_value}</IgnoreUnresolvedVariables>
<AssignTo createNew="true" transport="http" type="request"></AssignTo>
</AssignMessage>

```

Elements & Attributes

Description

AssignTo (Optional)

Specifies the variable to which the message will be assigned.

If this element is missing, it is treated as request or response, depending on the Flow to which the policy is attached. If attached to a response Flow, for example, the default type is response.

In some cases, you cannot change the object on which this policy works. For example, you cannot change query parameters or form parameters on the response, but can only do so on the request.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt '>
  <IgnoreUnresolvedVariables>[true|
false]</IgnoreUnresolvedVariables>
  <AssignTo createNew="[true|
false]" transport="http" type="[request|
response]">destination_variable_name</
AssignTo>
</AssignMessage>
```

Sample Code

Example

The following example specifies that the target is the original request that will be sent to the target endpoint:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt '>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo
createNew="false" transport="http"
type="request">destination_variable_name</
AssignTo>
</AssignMessage>
```

Elements & Attributes	Description
createNew (Optional)	<p>It is a Boolean value which indicates if the request or response object should be newly created or if the existing message should be modified.</p> <p>If the value is true, the policy creates a new request or response object, based on the type specified. If no name is specified for the new variable, the policy creates a new request or response object, based on the value of type.</p> <div data-bbox="707 577 1394 730" style="background-color: #f0f0f0; padding: 5px;"> <p>Note</p> <p>When a new request or response object is created, it deletes the existing one and replaces it completely.</p> </div> <p>If the value is false, the policy responds in one of the following ways:</p> <ul style="list-style-type: none"> • If the variable name to a request or response is resolved, the processing continues. • If the variable name to a request or response is not resolved, or is resolved to a non-message type, the policy throws an error. <p>If the value of createNew is not specified, the policy responds in one of the following ways:</p> <ul style="list-style-type: none"> • If createNew resolves to a message, the processing continues. • If createNew is not resolved, or is resolved to a non-message type, a new variable of type specified in type is created.
transport (Optional)	<p>It is a string which indicates the transport method for request and response messages. The only supported value is HTTP.</p>
type (Optional)	<p>It is a string that specifies the type of the new message, when createNew is true.</p> <p>Valid values: <code>request</code> or <code>response</code></p> <p>Default value: <code>request</code></p>

Elements & Attributes

Description

IgnoreUnresolvedVariables (Optional)

If **IgnoreUnresolvedVariables** is set to **false** and any variable cannot be resolved, then the policy throws an error.

If it is set to **true** and any variable is unresolvable, the variable is treated as empty string (Null).

Valid values: true or false

Default value: false

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <IgnoreUnresolvedVariables>[true|
false]</IgnoreUnresolvedVariables>
</AssignMessage>
```

↔ Sample Code

Example

The following example sets **IgnoreUnresolvedVariables** to **true**:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Copy source="response" >
    ...
  <IgnoreUnresolvedVariables>true</
IgnoreUnresolvedVariables>
</Copy>
</AssignMessage>
```

Copy (Optional)

source

Copies the specified information from the <source> to the variable specified in the <AssignTo> element.

If the source is not specified, it is treated as a simple message. If the source variable cannot be resolved, or resolves to a non-message type, <Copy> fails to respond.

Headers

Copies HTTP headers.

To copy multiple headers, mention the header name in the name attribute as below:

```
<Copy source='request'>
<Headers>
<Header name="headerA" />
<Header name="headerB" />
</Headers>
</Copy>
```

To copy all headers, specify `<Copy><Headers /></Copy>`.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Copy source="request|response">
    <!-- Can also be an empty array
    (<Headers />) -->
    <Headers>
      <Header
name="header_name">header_value</Header>
      ...
    </Headers>
  </Copy>
  <IgnoreUnresolvedVariables>[true|false]</
IgnoreUnresolvedVariables>
</AssignMessage>
```

Sample Code

Example

The following example copies the temp header from the request to the new CustomReq object:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Copy source="request">
    <Headers>
      <Header name="temp" />
    </Headers>
  </Copy>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
```

Elements & Attributes**Description**

```
<AssignTo  
createNew="true" transport="http"  
type="request">CustomReq</AssignTo>  
</AssignMessage>
```

QueryParams

Copies query parameters.

Note that the QueryParams is copied only when both the source and AssignTo type are request.

To copy all query parameters, specify `<Copy><QueryParams /></Copy>`.

You can use query parameters only when the message type is Request and the HTTP verb is GET.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt '>
  <Copy source="request|response">
    <!-- Can also be an empty array
    (<QueryParams/>) -->
    <QueryParams>
      <QueryParam
name="queryparam_name">queryparam_value</
QueryParam>
      . . .
    </QueryParams>
  </Copy>
<IgnoreUnresolvedVariables>[true|false]</
IgnoreUnresolvedVariables>
</AssignMessage>
```

Sample Code

Example

The following example copies the `temp_param` query parameter from the request into a new CustomReq object:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt '>
  <Copy source="request">
    <QueryParams>
      <QueryParam name="temp_param"/>
    </QueryParams>
  </Copy>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo
createNew="true" transport="http"
type="request">CustomReq</AssignTo>
</AssignMessage>
```


FormParams Copies the form parameters from the request specified in the <source> attribute of <Copy> to the request specified by AssignTo.

Note that the FormParams is copied only when the contentType is source and AssignTo is application/x-www-form-urlencoded.

To copy all form parameters, specify <Copy><FormParams /></Copy>.

You can use query parameters only when the message type is Request and the HTTP verb is POST. Additionally, you should meet one (or both) of the following criteria:

- Set the Form data to some value, or '' (empty string). For example, with **curl**, add **-d ""** to your request.
- Set the **Content-Length** header to if there is no data in the original request; otherwise, set it to the current length in bytes. For example, with **curl**, add **-H "Content-Length: 0"** to your request.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt'>
  <Copy source="[request|response]">
    <!-- Can also be an empty array
    (<FormParams/>) -->
    <FormParams>
      <FormParam
name="formparam_name">formparam_value</
FormParam>
      . . .
    </FormParams>
  </Copy>
</AssignMessage>
```

Sample Code

Example

The following example copies three form parameters to the custom request **CustomReq**:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
  <Copy source="request">
    <FormParams>
      <FormParam name="pName1" />
      <FormParam name="pName2" />
      <FormParam name="pName3" />
    </FormParams>
  </Copy>
</AssignMessage>
```

```

</FormParams>
</Copy>
<IgnoreUnresolvedVariables>>false</
IgnoreUnresolvedVariables>
  <AssignTo
createNew="true" transport="http"
type="request">CustomReq</AssignTo>
</AssignMessage>

```

Payload

Valid values: true or false.

If true, the Content-Type header is copied.

↔ Sample Code

Syntax

```

<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Copy source="[request|response]" >
    <Payload>[false|true]</Payload>
  </Copy>
</AssignMessage>>

```

↔ Sample Code

Example

The following example sets <Payload> to "true" so that the request payload is copied from the request to the response:

```

<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Copy source="request" >
    <Payload>true</Payload>
  </Copy>
  <IgnoreUnresolvedVariables>>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="true"
transport="http" type="response" />
</AssignMessage>

```

Version

Valid values: true or false.

If true, the HTTP version is copied.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Copy source="[request|response]" >
    <Version>[false|true]</Version>
  </Copy>
</AssignMessage>
```

Sample Code

Example

The following example sets <Version> to "true" on the request, which copies the version from the default request object to a new custom request object:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Copy source="request" >
    <Version>>true</Version>
  </Copy>
  <AssignTo
createNew="true" transport="http"
type="request">CustomReq</AssignTo>
</AssignMessage>
```

Verb

Valid values: true or false.

If true, the verb of the request gets assigned to the new request message, which is indicated by the AssignTo variable. This element is applicable only for HTTP request and not for response.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Copy source="[request|response]" >
    <Verb>[false|true]</Verb>
  </Copy>
</AssignMessage>
```

↔ Sample Code

Example

The following example sets <Verb> to "true", which copies the verb from the default request to a new custom request:

```
<AssignMessage name="copy-verb-1" >
  <Copy source="request" >
    <Verb>true</Verb>
  </Copy>
  <AssignTo
createNew="true" transport="http"
type="request">MyCustomRequest</AssignTo>
</AssignMessage>
```

Path

If true, the path of the request gets assigned to the path of the new request object, which is indicated by the AssignTo variable. This element is applicable only for HTTP request and not for response.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Copy source="[request|response]" >
    <Path>[false|true]</Path>
  </Copy>
</AssignMessage>
```

↔ Sample Code

Example

The following example indicates that Assign Message should copy the path from the source request to the new custom request object:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Copy source="request" >
    <Path>true</Path>
  </Copy>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo
createNew="true" transport="http"
type="request">CustomReq</AssignTo>
</AssignMessage>
```

StatusCode Valid values: `true` or `false`. If `true`, the response status gets assigned to the new response message, which is indicated by the `AssignTo` variable. This element is applicable only for HTTP request and not for response.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Copy source="[request|response]" >
    <StatusCode>[false|true]</StatusCode>
  </Copy>
</AssignMessage>
```

↔ Sample Code

Example

The following example sets `<StatusCode>` to `"true"`, which copies the status code from the default response object to a new custom response object:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Copy source="response" >
    <StatusCode>true</StatusCode>
  </Copy>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo
createNew="true" transport="http"
type="response">CustomReq</AssignTo>
</AssignMessage>
```

Reason
Phrase

If true, the reason phrase of the response gets assigned to the new response message, which is indicated by the AssignTo variable. This element is applicable only for HTTP request and not for response.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt'>
  <Copy source="[request|response]">
    <ReasonPhrase>[false|true]</
ReasonPhrase>
  </Copy>
</AssignMessage>
```

↔ Sample Code

Example

The following example sets <ReasonPhrase> to "true", which causes <Copy> to copy the reason phrase from the default response to a custom response object:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
  <Copy source="response">
    <ReasonPhrase>true</ReasonPhrase>
  </Copy>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo
createNew="true" transport="http"
type="response">CustomReq</AssignTo>
</AssignMessage>
```

Elements & Attributes

Description

Remove (Optional)

Headers

Removes HTTP headers from the variable specified in the <AssignTo>element. To remove all the headers, specify <Remove><Headers/></Remove>.

To remove specific headers, provide the header name in the name attribute as below:

```
<Remove>
<Headers>
<Header name="headerA" />
<Header name="headerB" />
</Headers>
</Remove>
```

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt'>
  <Remove>
    <!-- Can also be an empty array
    (<Headers/>) -->
    <Headers>
      <Header
name="header_name">header_value</Header>
      ...
    </Headers>
  </Remove>
</AssignMessage>
```

Sample Code

Example

The following example removes the **temp** header from the request:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
  <Remove>
    <Headers>
      <Header name="temp" />
    </Headers>
  </Remove>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="false"
transport="http" type="request" />
```



```
</AssignMessage>
```

QueryParams Removes the query parameters.

Note that the QueryParams are removed only when the AssignTo type is request and the HTTP verb is GET. To remove all query parameters, specify `<Remove><QueryParams /></Remove>`.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt'>
  <Remove>
    <!-- Can also be an empty array
    (<QueryParams />) -->
    <QueryParams>
      <QueryParam
name="queryparam_name">queryparam_value</
QueryParam>
      . . .
    </QueryParams>
  </Remove>
</AssignMessage>
```

Sample Code

Example

The following example removes all query parameters from the request:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
  <Remove>
    <QueryParams />
  </Remove>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="false"
transport="http" type="request" />
</AssignMessage>
```

FormParams Removes the form parameters.

Note that the FormParams are removed only when the contentType of AssignTo is application/x-www-form-urlencoded. To remove all query parameters, specify `<Remove><FormParams /></Remove>`.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt'>
  <Remove>
    <!-- Can also be an empty array
    (<FormParams/>) -->
    <FormParams>
      <FormParam
name="formparam_name">formparam_value</
FormParam>
      . . .
    </FormParams>
  </Remove>
</AssignMessage>
```

↔ Sample Code

Example

The following example removes three form parameters from the request:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
  <Remove>
    <FormParams>
      <FormParam name="form_param_1"/>
      <FormParam name="form_param_2"/>
      <FormParam name="form_param_3"/>
    </FormParams>
  </Remove>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="false"
transport="http" type="application/x-www-
form-urlencoded"/>
</AssignMessage>
```

Payload

Valid values: true or false. If true, the payload is cleared.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Remove>
    <Payload>[false|true]</Payload>
  </Remove>
</AssignMessage>
```

↔ Sample Code

Example

The following example sets <Payload> to "true" so that the request payload is cleared:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Remove>
    <Payload>true</Payload>
  </Remove>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="false"
transport="http" type="request" />
</AssignMessage>
```

Elements & Attributes

Description

Add (Optional)

Headers

Adds the headers in the variable specified in the <AssignTo> element.

Note that the empty header <Add><Headers/></Add> does not add any header. The same holds true for QueryParams and FormParams.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Add>
    <Headers>
      <Header
name="header_name">header_value</Header>
      ...
    </Headers>
  </Add>
</AssignMessage>
```

Sample Code

Example

The following example adds the temp header to the request message, and assigns the value of the request.temp flow variable to that header:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Add>
    <Headers>
      <Header name="temp">{request.temp}</
Header>
    </Headers>
  </Add>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="false"
transport="http" type="request"/>
</AssignMessage>
```

QueryParams Adds the query parameters.

You can use query parameters only when the message type is Request and the HTTP verb is GET.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Add>
    <QueryParams>
      <QueryParam
name="queryparam_name">queryparam_value</
QueryParam>
      . . .
    </QueryParams>
  </Add>
</AssignMessage>
```

Sample Code

Example

The following example adds the query parameter `tempParam` to the request and assigns the value `82` to it:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Add>
    <QueryParams>
      <QueryParam name="tempParam">82</
QueryParam>
    </QueryParams>
  </Add>
  <IgnoreUnresolvedVariables>>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="false"
transport="http" type="request" />
</AssignMessage>
```

FormParams Adds the form parameters and the contentType of message is changed to application/x-www-form-urlencoded.

You can use form parameters only when the message type is Request and the HTTP verb is POST. Additionally, you should meet one (or both) of the following criteria:

- Set the Form data to some value, or '' (empty string). For example, with **curl**, add **-d ""** to your request.
- Set the **Content-Length** header to if there is no data in the original request; otherwise, set it to the current length in bytes. For example, with **curl**, add **-H "Content-Length: 0"** to your request.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Add>
    <FormParams>
      <FormParam
name="formparam_name">formparam_value</
FormParam>
      ...
    </FormParams>
    <AssignTo createNew="[true|false]"
transport="http"
type="[request|
response]">destination_variable_name</
AssignTo>
  </Add>
</AssignMessage>
```

Sample Code

Example

The following example adds a single form parameter ("**answer**") and a static value ("**42**") to the request:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Add>
    <FormParams>
      <FormParam name="answer">42</
FormParam>
    </FormParams>
  </Add>
  <IgnoreUnresolvedVariables>>false</
IgnoreUnresolvedVariables>
```

Elements & Attributes

Description

```
<AssignTo transport="http"
type="request"></AssignTo>
</AssignMessage>
```

Set (Optional)

Headers

Sets or overwrites the HTTP headers in the variable specified in the <AssignTo> element.

Note that the empty header <Set><Headers/></Set> does not set any header. The same holds true for QueryParams and FormParams.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt'>
  <Set>
    <Headers>
      <Header
name="header_name">header_value</Header>
      ...
    </Headers>
  </Set>
</AssignMessage>
```

Sample Code

Example

The following example sets the user-agent header to the value of the request.header.user-agent variable:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
  <Headers>
    <Header
name="user-agent">{request.header.user-
agent}</Header>
  </Headers>
  <IgnoreUnresolvedVariables>>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="true"
transport="http" type="response"/>
</AssignMessage>
```

QueryParams Sets or overwrites the query parameters in a request.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Set>
    <QueryParams>
      <QueryParam
name="queryparam_name">queryparam_value</
QueryParam>
      ...
    </QueryParams>
  </Set>
</AssignMessage>
```

↔ Sample Code

Example

The following example sets the "temp" query parameter to the value of the request.header.temp variable:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <QueryParams>
    <QueryParam
name="temp">{request.header.temp}</
QueryParam>
  </QueryParams>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
</AssignMessage>
```


FormParams Sets or overwrites the form parameters and the contentType of message changes to application/x-www-form-urlencoded.

You can use form parameters only when the message type is Request and the HTTP verb is POST.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Set>
    <FormParams>
      <FormParam
name="formparam_name">formparam_value</
FormParam>
      ...
    </FormParams>
  </Set>
</AssignMessage>
```

Sample Code

Example

The following example sets a form parameter called "tempparam" to the value of the request.header.tempparam variable in a new custom request:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <FormParams>
    <FormParam
name="tempparam">{request.header.tempparam
}</FormParam>
  </FormParams>
  <IgnoreUnresolvedVariables>>false</
IgnoreUnresolvedVariables>
  <AssignTo
createNew="true" transport="http"
type="request">CustomReq</AssignTo>
</AssignMessage>
```

Payload

Enter the payloads of type json, xml, plain text, and so on, within this element.

Following are the attributes (optional) of <Payload>:

- **contentType**: It is a string which if specified, assigns the value of **contentType** to the Content-Type HTTP header.
- **variablePrefix**: It is a character which optionally specifies the leading delimiter on a flow variable. The default is "{".
- **variableSuffix**: It is a character which optionally specifies the trailing delimiter on a flow variable. The default is "}

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Set>
    <Payload contentType="content_type"
variablePrefix="prefix"
variableSuffix="suffix">new_payload</
Payload>
  </Set>
</AssignMessage>
```

Sample Code

Example

The following example sets a JSON payload:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Set>
    <Payload contentType="application/
json">
      {"name":"foo", "type":"bar"}
    </Payload>
  </Set>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
</AssignMessage>
```

Note

If the payload content is of the type XML, and gets changed unexpectedly during API proxy execution, please wrap it in <![CDATA[...]]> element.

This way, you can ensure that the payload content is treated as a string and thereby it is not getting processed by the API Proxy back-end system as XML.

You can still process the dynamic data like query parameters or variables within the content wrapped in CDATA.

↗ Sample Code

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
  <Set>
    <Payload
contentType="text/xml"><![CDATA[
.
. <!--xml content
here
.
]]>
</Payload>
</Set>
</AssignMessage>
```

Version

Sets the HTTP version on a request.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <Set>
    <Version>[1.0|1.1|{variable}]</Verb>
  </Set>
</AssignMessage>
```

↔ Sample Code

Example

The following example uses a variable in curly braces to set the version number:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Set>
    <Version>{my_version}</Version>
  </Set>
  <IgnoreUnresolvedVariables>>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="true"
transport="http" type="request" />
</AssignMessage>
```

The content of <Version>, wrapped in curly braces is a message template and is replaced at runtime with the value of the referenced variable.

Reason
Phrase

Sets the reason phrase only when AssignTo type is response. This is generally used in combination with the status code for debugging.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"  
continueOnError="[true|false]"  
enabled="[true|false]" xmlns='http://  
www.sap.com/apimgmt'>  
  <Set>  
    <ReasonPhrase>reason_for_error or  
{variable}</ReasonPhrase>  
  </Set>  
</AssignMessage>
```

↔ Sample Code

Example

The following example uses a variable to populate a reason phrase:

```
<AssignMessage async="false"  
continueOnError="false" enabled="true"  
xmlns='http://www.sap.com/apimgmt'>  
  <Set>  
  
  <ReasonPhrase>{calloutresponse.reason.phra  
se}</ReasonPhrase>  
  </Set>  
  <IgnoreUnresolvedVariables>>false</  
IgnoreUnresolvedVariables>  
  <AssignTo createNew="true"  
transport="http" type="response"/>  
</AssignMessage>
```

The content of <ReasonPhrase>, wrapped in curly braces is a message template and is replaced at runtime with the value of the referenced variable.

Status Code Sets the status code only when AssignTo type is response.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt'>
  <Set>
    <StatusCode>HTTP_status_code or
{variable}</StatusCode>
  </Set>
</AssignMessage>
```

↔ Sample Code

Example

The following example uses a variable to populate a status code:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
  <Set>

<StatusCode>{calloutresponse.status.code}<
/StatusCode>
  </Set>
  <IgnoreUnresolvedVariables>>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="true"
transport="http" type="response"/>
</AssignMessage>
```

The content of <StatusCode>, wrapped in curly braces is a message template and is replaced at runtime with the value of the referenced variable.

Verb Sets the HTTP verb and path only when AssignTo type is request.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt'>
  <Set>
    <Verb>[GET|POST|PUT|PATCH|DELETE|
{variable}]</Verb>
  </Set>
</AssignMessage>
```

Elements & Attributes**Description**

Path

↔ Sample Code

Example

The following example uses a variable to populate a verb:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <Set>
    <Verb>{my_variable}</Verb>
  </Set>
  <IgnoreUnresolvedVariables>>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="true"
transport="http" type="request" />
</AssignMessage>
```

The content of <Verb>, wrapped in curly braces is a message template and is replaced at runtime with the value of the referenced variable.

Elements & Attributes

Description

AssignVariable

Name (Required)

It is a string that specifies the name of the variable. If the variable named in **AssignVariable** does not exist, the policy creates one with that name.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt'>
  <AssignVariable>
    <Name>variable_name</Name>
  </AssignVariable>
</AssignMessage>
```

Sample Code

Example

The following example specifies the destination variable as **var** and sets it to the value **"83"**:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
  <AssignVariable>
    <Name>var</Name>
    <Value>83</Value>
  </AssignVariable>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="true"
transport="http" type="request" />
</AssignMessage>
```

If **myvar** does not exist, **AssignVariable** creates it.

Ref (Optional) Reference that assigns value (as a flow variable and not a string variable) to the variable.

If you want to assign a literal string value to the variable, use the **Value** element instead.

- Do this (no brackets): `<Ref>client.host</Ref>`
- Do NOT do this (brackets): `<Ref>{client.host}</Ref>`

Define the default value for the destination flow variable by using `<Value>` along with `<Ref>`. If the flow variable specified by `<Ref>` does not exist, is not readable, or is null, then the value of `<Value>` is assigned to the destination flow variable instead.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <AssignVariable>
    <Name>variable_name</Name>
    <Ref>source_variable</Ref>
  </AssignVariable>
</AssignMessage>
```

Sample Code

Example

The following example assigns the value of the flow variable `request.header.temp` to the destination flow variable `var` and the value of the query parameter `test` to the `test` variable:

```
<AssignMessage async="false"
continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt' >
  <AssignVariable>
    <Name>var</Name>
    <Ref>request.header.temp</Ref>
  </AssignVariable>
  <AssignVariable>
    <Name>test</Name>
    <Ref>request.queryparam.test</Ref>
  </AssignVariable>
  <IgnoreUnresolvedVariables>false</
IgnoreUnresolvedVariables>
  <AssignTo createNew="true"
transport="http" type="request" />
</AssignMessage>
```

Template (Optional)

It is a string that specifies the message template, which support functions like escaping and case conversion.

↔ Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt'>
  <AssignVariable>
    <Template>message_template</Template>
  </AssignVariable>
</AssignMessage>
```

↔ Sample Code

Example

The following example uses the message template syntax to concatenate two context variables with a literal string (hyphen) between them:

```
<AssignMessage name='template-1'>
  <IgnoreUnresolvedVariables>>false</
IgnoreUnresolvedVariables>
  <AssignVariable>
    <Name>my_destination_variable</Name>
    <Value>BADDBEEF</Value>
    <Template>{system.uuid}-{messageid}</
Template>
  </AssignVariable>
</AssignMessage>
```

Elements & Attributes

Description

Value (Optional)

It is a string that specifies the value of the variable.

If you use a combination of the <Value> and <Ref> elements, <Value> acts as default. If <Ref> is not specified or is unresolvable, this literal string value is assigned to the variable.

Sample Code

Syntax

```
<AssignMessage async="false|true"
continueOnError="[true|false]"
enabled="[true|false]" xmlns='http://
www.sap.com/apimgmt' >
  <AssignVariable>
    <Name>variable_name</Name>
    <Value>variable_value</Value>
  </AssignVariable>
</AssignMessage>
```

Sample Code

Example

The following example assigns the value of the flow variable **request.header.temp** to the destination flow variable **var** and the value of the query parameter **test** to the **test** variable:

```
<AssignMessage name="assignvariable-2">
  <AssignVariable>
    <Name>var</Name>
    <Value>ErrorOnCopy</Value>
    <Ref>request.header.temp</Ref>
  </AssignVariable>
  <AssignVariable>
    <Name>test</Name>
    <Value>ErrorOnCopy</Value>
    <Ref>request.queryparam.test</Ref>
  </AssignVariable>
</AssignMessage>
```

If either assignment fails, the value "ErrorOnCopy" is assigned to the variable.

During the policy execution, the following errors can occur:

Error Cause

Error Name	Cause
UnresolvedVariable	A flow variable referenced in the policy does not exist. Be sure that the variable is in scope - some of the built-in variables are only available in certain flow contexts.

Error Name	Cause
VariableOfNonMsgType	The policy tried to assign a value to a non-message type variable. Message type variables include request and response. They also can be custom variables that are of type message. You might see this in the <Copy> element if you set the source attribute to a variable that is not of type Message.
SetVariableFailed	The policy was not able to set a variable. See the fault string for the name of the unresolved variable
InvalidIndex	The index must be greater than zero when specified in the Copy and Remove operations. For example, a query parameter can have multiple values. This error occurs if you specify an invalid index, such as 0 or a negative number.
InvalidVariableName	The policy schema validation failed because a variable name is invalid.
InvalidPayload	A payload specified in the policy is invalid.

Following fault variables are set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	The [prefix] is assignmessage. The [policy_name] is the name of the policy that threw the error.	assignmessage.AM-SetResponse.failed = true
fault.name = [error_name]	[error_name] is the specific error name to check for as listed in the table above.	fault.name = "UnresolvedVariable"

Following is an example of an error response:

Sample Code

```
{
  "fault": {
    "detail": {
      "errorcode": "steps.assignmessage.VariableOfNonMsgType"
    },
    "faultstring": "AssignMessage[AM-SetResponse]: value of variable is not
of type Message"
  }
}
```

Following is an example of a fault rule:

Sample Code

```
<faultrule name="VariableOfNonMsgType"></faultrule><FaultRule name="Assign
Message Faults">
  <Step>
    <Name>AM-CustomNonMessageTypeErrorResponse</Name>
    <Condition>(fault.name Matches "VariableOfNonMsgType") </Condition>
  </Step>
</Step>
```

```

<Name>AM-CustomSetVariableErrorResponse</Name>
<Condition>(fault.name = "SetVariableFailed")</Condition>
</Step>
<Condition>(assignmessage.failed = true) </Condition>
</FaultRule>

```

Related Information

[Access Entity \[page 212\]](#)

1.5.1.4.1.4 Basic Authentication

Basic authentication policy takes a username and password, encode them to Base64 format and writes the resulting value to a variable. The resulting value is typically written to an HTTP header, such as the Authorization header in the form `Basic Base64EncodeString`.

Note

This policy does not enforce basic authentication on a request to an API proxy. Instead, you use it to Base64 encode or decode credentials, typically when connecting to a backend server or using a service callout policy that requires basic authentication.

The policy has two modes of operations:

- Encode: Base64 encodes a username and password stored in variables
- Decode: Decodes the username and password from a Base64 encoded string

The username and password are commonly stored the key/value store and then read from the key/value store at runtime.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

Code Syntax

```

<!-- Use Case: Create and set authorization header for the current request
from the given user name and password.
The policy retrieves user name and password from the request body which is
supplied as form parameters.
-->

```

```

<BasicAuthentication async='true' continueOnError='false' enabled='true'
xmlns='http://www.sap.com/apimgmt'>
  <Operation>Encode</Operation>
  <IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
  <User ref='request.formparam.username'></User>
  <Password ref='request.formparam.password'></Password>
  <AssignTo createNew="false">request.header.Authorization</AssignTo>
</BasicAuthentication>
<!-- Use case: Extract the user credentials from the authorization header
User name and password is extracted from the authorization header of the
incoming request.
The user name and password is set into the flow variables named as
"current.username" and "current.password" respectively.
-->
<BasicAuthentication async='true' continueOnError='false' enabled='true'
xmlns='http://www.sap.com/apimgmt'>
  <Operation>Decode</Operation>
  <IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
  <User ref='current.username'></User>
  <Password ref='current.password'></Password>
  <Source>request.header.Authorization</Source>
</BasicAuthentication>

```

Elements and Attributes	Description
Operation (Mandatory)	Supports values <code>Encode</code> or <code>Decode</code> . This setting will enable you to encode credentials to populate an HTTP header on an outbound request, or decode encoded credentials from HTTP header of an inbound request.
IgnoreUnresolvedVariables (Optional)	Supports values <code>true</code> or <code>false</code> . This setting determines whether to throw an error if the variables defined in the policy is not resolved. If set to <code>true</code> , the policy will not throw an error if a variable cannot be resolved. In basic authentication policy, it is recommended to set this value to <code>false</code> , because it is beneficial to throw an error if a username or password cannot be found in the variables specified.
User ref (Mandatory)	Settings for username. For encoding, set a reference attribute to the username to dynamically retrieve value from a variable. For decoding, specify the flow variable in which the decoded username is to be placed.
Password ref (Mandatory)	Settings for password. For encoding, set a reference attribute to the password to dynamically retrieve the value from a variable. For decoding, specify the flow variable in which the decoded password is to be placed.
AssignTo	Assigns the encoded value of username and password to a variable. Do not use this if the operation is <code>Decode</code> .
Source	The encoded value of username and password is retrieved from Source. Do not use this if the operation is <code>Encode</code> .

During the policy execution, the following errors can occur:

Error Cause

Error Name	Cause
UnresolvedVariable	The required source variables for the decode or encode are not present. This error can only occur if IgnoreUnresolvedVariables is false.
InvalidBasicAuthenticationSource	On a decode when the incoming Base64 encoded string does not contain a valid value or the header is malformed (for example does not start with "Basic").
UserNameRequired	The <User> element must be present for the named operation. See the fault string.
PasswordRequired	The <Password> element must be present for the named operation. See the fault string.
AssignToRequired	The <AssignTo> element must be present for the named operation. See the fault string.
SourceRequired	The <Source> element must be present for the named operation. See the fault string.

Following fault variables are set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	The [prefix] is BasicAuthentication. The [policy_name] is the name of the policy that threw the error.	BasicAuthentication.BA-Authenticate.failed = true
fault.[error_name]	[error_name] = The specific error name to check for as listed in the table above.	fault.name Matches "UnresolvedVariable"

Related Information

[Key Value Map Operations \[page 321\]](#)

1.5.1.4.1.5 Concurrent Rate Limit

The Concurrent Rate Limit policy is being decommissioned. The support for the Concurrent Rate Limit policy has come to an end. You can no longer create or update an API proxy with Concurrent Rate Limit policy. If you're still using the policy and wondering which policy to use to best meet your rate-limiting needs, see [Replace Concurrent Rate Limit Policy with Alternative Policies \[page 256\]](#).

Related Information

[Quota \[page 336\]](#)

[Spike Arrest \[page 353\]](#)

1.5.1.4.1.6 Replace Concurrent Rate Limit Policy with Alternative Policies

The Concurrent Rate Limit policy was designed to cater to slow backend systems, however, the architecture used by it affected the performance of the APIs. Therefore, this policy has been **decommissioned**.

Note

You can use the Spike Arrest policy to limit the number of requests to the backend systems over a specified period of time. You can also use Quota policy to limit the number of request messages that an API proxy allows over a period of time, such as minute, hour, day, week, or month. The Quota policy shouldn't be used to protect target backend systems against traffic spikes.

If you're wondering which policy to use to best meet your rate limiting needs, refer the following comparison chart:

Comparison Between the Quota, Spike Arrest, and Concurrent Rate Limit Policy

Quota	Spike Arrest	Concurrent Rate Limit (Decommissioned)
Use it to limit the number of connections apps can make to your API proxy's target backend over a specific period of time.	Use it to protect your API proxy's target backend against severe traffic spikes and denial of service attacks.	Use it to limit the number of concurrent connections apps can make to your API proxy's target backend.
Don't use it to protect your API proxy's target backend against traffic spikes. Use the Spike Arrest policy instead.	Don't use it to count and limit the number of connections apps can make to your API proxy's target backend over a specific period of time. Use the Quota policy instead.	Don't use it to limit the number of connections applications can make to your API proxy's target backend over a specific period of time. Use the Quota policy for this purpose.
The Quota policy stores the count.	The Spike Arrest policy doesn't store the count.	The Concurrent Rate Limit policy stores the count.
Attach the policy to the ProxyEndpoint Request Pre-Flow , generally after the authentication of the user. This enables the policy to check the quota counter at the entry point of your API proxy.	Attach the policy to the ProxyEndpoint Request Pre-Flow , generally at the very beginning of the flow. This provides spike protection at the entry point of your API proxy.	This policy must be attached in these three locations: <ul style="list-style-type: none">• TargetEndpoint Request Pre-Flow• TargetEndpoint Response Pre-Flow• TargetEndpoint DefaultFaultRule

Quota	Spike Arrest	Concurrent Rate Limit (Decomissioned)
HTTP status code when limit has been reached: 500 (Internal Server Error) *	HTTP status code when limit has been reached: 500 (Internal Server Error) *	HTTP status code when limit has been reached: 503 (Service Unavailable)
<p>Good to know facts:</p> <ul style="list-style-type: none"> Quota counter is stored in Cassandra. Configure the policy to synchronize the counter asynchronously to save resources. Asynchronous counter synchronization can cause a delay in the rate limiting response, which allows calls slightly in excess of the limit you've set. <p>See Quota [page 336] for more information.</p>	<p>Good to know facts:</p> <ul style="list-style-type: none"> Performs throttling based on the time at which the last traffic was received. This time is stored per message processor. If you specify a rate limit of 100 calls per second, only 1 call every 1/100 second (10 ms) is allowed on the message processor. A second call within 10 ms is rejected. Even with a high rate limit per second, nearly simultaneous requests results in rejections. <p>See Spike Arrest [page 353] for more information.</p>	<p>Good to know facts:</p> <ul style="list-style-type: none"> Keeps a count of concurrent connections per message processor. While an individual API proxy handles just a few connections collectively, the connections to a set of replicated API proxies pointing to the same backend service swamp the capacity of the service. Use this policy to limit this traffic to a manageable number of connections. This policy is known to slow performance in API proxies that handle a high number of transactions per second (TPS). For high-TPS API proxies, if ConcurrentRateLimit slows performance to unacceptable levels, try using SpikeArrest instead. <p>See Concurrent Rate Limit [page 255] for more information.</p>

❁ Example

Refer this example to replace Concurrent Rate Limit policy with Spike Arrest policy.

In Concurrent Rate Limit, four concurrent connections were allowed. In Spike Arrest, four requests per minute are allowed. You can alter the Spike Arrest Policy based on the number of requests allowed for your backend system.

🔗 Sample Code

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<SpikeArrest async="false" continueOnError="false" enabled="true" name="Spike-Arrest-1">
  <DisplayName>Spike-Arrest-1</DisplayName>
  <Properties/>
  <Rate>4pm</Rate>
  <UseEffectiveCount>true</UseEffectiveCount></SpikeArrest>
```

🔗 Sample Code

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```

<ConcurrentRatelimit async="false" continueOnError="false" enabled="true"
name="Concurrent Rate Lmit-1">
<DisplayName>Concurrent Rate Lmit-1</DisplayName>
  <AllowConnections count="4" ttl="7"/>
  <Distributed>true</Distributed>
  <StrictOnTtl>true</StrictOnTtl>
  <TargetIdentifier name="" />
  <UseEffectiveCount>true</UseEffectiveCount>
</ConcurrentRatelimit>

```

Note

Concurrent Rate Limit and the Spike Arrest policy aren't the same and works differently. However, you can tune it to allow specific requests per second, and set the maximum number of requests your backend system receives.

1.5.1.4.1.7 Extract Variables

The Extract variables policy can be used to extract content from the HTTP request or response messages of the API Proxy and assign that content to specific variables that can be accessed during the execution of the API Proxy.

For more information on how to extract different variables, see [Examples \[page 264\]](#).

This policy can be applied on the request or response stream of the proxy endpoint or target endpoint.

You can attach the policy in one of the following locations:

ProxyEndpoint				TargetEndpoint			←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

Code Syntax

```

<!-- The policy will extract data at xpath /Developer/Email and store in
variable "email" for xml payload -->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ExtractVariables async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
  <Source>AccessEntity.GetDeveloperProfile</Source>
  <XMLPayload>
    <Variable name="email" type="string">
      <XPath>/Developer/Email</XPath>
    </Variable>
  </XMLPayload>
</ExtractVariables>

```

Element	Attribute Name	Description
Pattern	ignoreCase	<p>Specifies the pattern to be applied to the parent element. The list of parent elements are:</p> <ul style="list-style-type: none"> • URIPath • QueryParam • FormParam • Header • Variable
<URIPath>		<p>Extracts the value of a variable from the specified URI path of the specified message. This element is applicable only if source is request.</p> <p>* You must include at least one of the following:</p> <p><URIPath>, <QueryParam>, <Header>, <FormParam>, <JSONPayload>, or <XMLPayload></p>
<QueryParam>	name(Mandatory)	<p>Extracts the value of a variable from the specified query parameter of the specified message.</p> <p>* You must include at least one of the following:</p> <p><URIPath>, <QueryParam>, <Header>, <FormParam>, <JSONPayload>, or <XMLPayload></p>
<Header>	name(Mandatory)	<p>Extracts the value of a variable from the specified HTTP header of the specified message.</p> <p>* You must include at least one of the following:</p> <p><URIPath>, <QueryParam>, <Header>, <FormParam>, <JSONPayload>, or <XMLPayload></p>
<FormParam>	name(Mandatory)	<p>Extracts the value of a variable from the specified form parameter. Form parameters can be extracted only when the contentType of the specified message is application/x-www-form-urlencoded.</p> <p>* You must include at least one of the following:</p> <p><URIPath>, <QueryParam>, <Header>, <FormParam>, <JSONPayload>, or <XMLPayload></p>

Element	Attribute Name	Description
<Variable>	name(Mandatory)	<p>Specifies the name of the variable to which the extracted value will be assigned. If there is a VariablePrefix element, then this name will be appended, as variableprefix.name.</p> <p>For example, suppose the name is specified as Developer:</p> <p>If <VariablePrefix> is not specified, the extracted values are assigned to Developer.</p> <p>If <VariablePrefix> is specified as MyUser, the extracted values are assigned to MyUser.Developer.</p> <div style="border: 1px solid #ccc; background-color: #f9f9f9; padding: 10px; margin-top: 10px;"> <p>Note</p> <p>When an XML variable is not resolved via an XPath expression, the ExtractVariables policy will result in an error. So, continueOnError or IgnoreUnresolvedVariables should be set to true to allow the execution of the policy.</p> </div>
<IgnoreUnresolvedVariables>		Set to true to treat any unresolvable variable as an empty string (null). Set to false if you want the policy to throw an error when any referenced variable is unresolvable.
<JSONPayload>		<p>Specifies the JSON formatted message from which the value of the variable will be extracted.</p> <p>* You must include at least one of the following:</p> <p><URIPath>, <QueryParam>, <Header>, <FormParam>, <JSONPayload>, or <XMLPayload></p>
<JSONPayload><Variable>		Specific the variable where the extracted value will be assigned.
<JSONPayload><Variable><JSONPath>		<p>Specifies the JSON path used to extract the value of a variable from a JSON formatted message.</p> <p>JSON payloads are extracted only when the content-Type of the specified message is application/json.</p>

Element	Attribute Name	Description
<Source>		<p>Specifies the message to be parsed. The value of Source defaults to message. The message value is context-sensitive; In a request flow, message resolves to the request message.</p> <p>In a response flow, message resolves to the response message.</p> <p>If the source variable cannot be resolved, or resolves to a nonmessage type, the policy will fail to respond.</p> <p>In advanced configurations, source can resolve to a variable containing a message generated by another policy, such as one generated from a ServiceCallout.</p>
	clearPayload	<p>Set to true is you want to clear the request payload after the request is sent to the HTTP target.</p> <p>Use the clearPayload option only if the request message is not required after the ServiceCallout is executed because clearPayload allocates memory during message processing.</p>
<VariablePrefix>		The complete variable name is created by joining the <VariablePrefix>, a dot, and the name you define in curly braces in the Pattern element.
<XMLPayload>		<p>Specifies the XML formatted message from which the value of the variable will be extracted.</p> <p>* You must include at least one of the following:</p> <p><URIPath>, <QueryParam>, <Header>, <FormParam>, <JSONPayload>, or <XMLPayload></p>
	stopPayload Processing	Set to true to stop XPath evaluation after one variable is populated.
<XMLPayload> <Namespaces>		Specifies the namespace to be used in the XPath evaluation. XML payloads are extracted only when the contentType of the message is text/xml, application/xml, or application/*+xml

Note

When an XML variable is not resolved via an XPath expression, the ExtractVariables policy will result in an error. So, continueOnError or IgnoreUnresolvedVariables should be set to true to allow the execution of the policy.

Element	Attribute Name	Description
<XMLPayload><Variable>		Specifies variable to which the extracted value will be assigned.
	type	Specifies the data type of the variable value. Supported data types are as follows: <ul style="list-style-type: none"> • string • boolean • integer • long • float • double • nodeset (returns an XML fragment)
XPath		Specifies the XPath defined for the variable. Only XPath 1.0 expressions are supported.
<XMLPayload>		Specifies the XPath defined for the variable. Only XPath 1.0 expressions are supported.
<Variable>		
<XPath>		

During the policy execution, the following errors can occur:

Error Cause

Error Name	Cause
ExecutionFailed	The policy tried to parse input that is malformed or otherwise invalid. The policy tried to parse XML with a namespace that is not declared in the <Namespace> element.
SourceMessageNotAvailable	A variable specified in <Source> is out of scope or can't be resolved. For example, if the policy executes in the request flow, the <Source> variable cannot be set to response or error.
SetVariableFailed	The policy was not able to set a variable value.
ImmutableVariable	A variable used in the policy is immutable. The policy was unable to set this variable.
VariableResolutionFailed	The policy could not resolve a variable. Be sure the variable you are trying to set exists in the runtime flow.
UnsupportedOperation	The policy tried to perform an unsupported operation on named flow variables.
UnableToCast	The policy was unable to cast a variable.
JSONPathCompilationFailed	The policy was unable to compile a JSON path expression.
JsonPathParsingFailure	The policy was unable to parse a JSON path to extract data into flow variables.

Error Name	Cause
InvalidJSONPath	A JSON path used in the policy is invalid.
NothingToExtract	A required element is missing from the policy. At least one of these elements is required: URIPath, QueryParam, Header, FormParam, XMLPayload, JSONPayload.
NONEmptyPrefixMappedToEmptyURI	The <XMLPayload> element is not configured properly. A non-empty prefix cannot be mapped to an empty URI.
DuplicatePrefix	The <XMLPayload> element is not configured properly. There is a duplicate prefix.
NoXPathToEvaluate	There are no XPath expressions to evaluate. An <XPath> child element must be specified.
EmptyXPathExpression	The policy has invalid configuration of the <Variable> child element of an <XMLPayload> element.
NoJSONPathsToEvaluate	You do not specify a <JSONPath> child element where it is required.
EmptyJSONPathExpression	The policy has an empty child element <JSONPath> in a element <JSONPayload>.
MissingName	The name attribute is missing from a policy element that requires it.
PatternWithoutVariable	A <Pattern> element that does not have a variable specified. The element requires the name of the variable in which extracted data will be stored.
CannotBeConvertedToNodeset	The result of an XPath expression cannot be converted to type nodeset.
JSONPathCompilationFailed	The policy could not compile a specified JSON Path.
InstantiationFailed	The policy could not be instantiated.
XPathCompilationFailed	The policy could not compile a specified XPath. Be sure to declare any namespaces that are used in the XPath.
InvalidPattern	A <Pattern> element is invalid in one of these elements: URIPath, QueryParam, Header, FormParam, XMLPayload, JSONPayload.

Following fault variables are set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	The [prefix] is extractvariables. The [policy_name] is the name of the policy to check.	extractvariables.EV-ParseJsonResponse.failed = true
fault.[error_name]	[error_name] = The specific error name to check for as listed in the table above.	fault.name = "SourceMessageNotAvailable"

Related Information

[Examples \[page 264\]](#)

1.5.1.4.1.7.1 Examples

You can use the Extract variables policy to choose the names of the variables to be set.

You choose the source of the variable, and how many variables to extract and set. Below are some examples to illustrate how this policy works:

→ Remember

When an XML variable is not resolved via an XPath expression, the ExtractVariables policy will result in an error. So, continueOnError or IgnoreUnresolvedVariables should be set to true to allow the execution of the policy.

Extract a variable from a query parameter

Consider an example where your API design specifies that incoming requests must carry a query parameter named code, which holds a term that looks like ABCXXXXX, where ABC is fixed, and the XXXXX denotes a varying string.

Sample Code

```
<ExtractVariables>
  <QueryParam name="code">
    <Pattern ignoreCase="true">ABC{abccode}</Pattern>
  </QueryParam>
  <VariablePrefix>queryinfo</VariablePrefix>
  <IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
</ExtractVariables>
```

Say, after a GET call, API Management sets the variable queryinfo.abccode to the value XXXXX. After API Management executes this Extract Variables policy, subsequent policies, or code in the processing flow can refer to the variable named queryinfo.abccode to get the string value XXXXX (for example, 12345).

Extract variables from a JSON payload

The Extract Variables policy can also extract values from JSON messages. The example below illustrates how to extract a variable from a portion of a JSON message payload. Consider this response payload:

Sample Code

```
{
```



```
"results": [{
  "key1": "value1",
  "key2": "value2"
}]
}
```

An element in the Extract Variables policy tells it to extract from a JSON payload. Rather than using text patterns, as you would when extracting values from headers, URI paths or query parameters, with JSON specify the portion to extract via a JSON path expression in which the \$ character refers to the root node of the JSON. Here's an example policy to illustrate:

Sample Code

```
<ExtractVariables>
  <Source>response</Source>
  <VariablePrefix>myresp</VariablePrefix>
  <JSONPayload>
    <Variable name="first_key">
      <JSONPath>$.results[0].key1</JSONPath>
    </Variable>
  </JSONPayload>
</ExtractVariables>
```

With this policy applied to the above message, API Management will extract a variable called `myresp.first_key` which will contain the value `value1`.

Extract variables from an XML message

You can extract variables from an XML payload, using XPath and explicitly named variables. Consider the below payload:

Sample Code

```
<RootElement>
  <Element1>
    <Element2 attr="myattr"/>
  </Element1>
</RootElement>
```

Sample Code

```
<ExtractVariables>
  <Source>response</Source>
  <VariablePrefix>myprefix</VariablePrefix>
  <XMLPayload>
    <Variable name="attrvalue" type="string">
      <XPath>/RootElement/Element1/Element2/@attr</XPath>
    </Variable>
  </XMLPayload>
</ExtractVariables>
```

With this policy, SAP API Management will set a variable called `myprefix.attrvalue` with the value `myattr`.

1.5.1.4.1.8 Caching Policies

Caching policies describe how cached values are written, retrieved, and purged at runtime. It is also used to return cached responses between refreshes to decrease the number of requests reaching the backend.

Populate Cache [page 266]

An OAuth access token is written to the cache using a Populate Cache policy. The OAuth token is retrieved for subsequent requests by a Lookup Cache policy.

Lookup Cache [page 268]

An OAuth access token is written to the cache using a Populate Cache policy. The OAuth token is retrieved for subsequent requests by a Lookup Cache policy.

Invalidate Cache [page 270]

The cache can be invalidated explicitly by specifying an HTTP header. When a request that contains the specified HTTP header is received, the cache will be flushed.

Response Cache [page 272]

1.5.1.4.1.8.1 Populate Cache

An OAuth access token is written to the cache using a Populate Cache policy. The OAuth token is retrieved for subsequent requests by a Lookup Cache policy.

At runtime, the Populate Cache policy writes data from the variable you specified in the <Source> element to the cache you specified in the <CacheResource> element. You can use the <CacheKey>, <Scope>, and <Prefix> elements to specify a key that you can use from the Lookup Cache policy to retrieve the value. Use the <ExpirySettings> element to configure when the cached value should expire.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	←Response
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

Code Syntax

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<PopulateCache async="false" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
  <CacheKey>
    <KeyFragment ref="request.header.userid"></KeyFragment>
  </CacheKey>
  <Scope>Exclusive</Scope>
  <ExpirySettings>
    <TimeOfDay ref="time_variable">expiration_time</TimeOfDay>
    <TimeoutInSec ref="duration_variable">seconds_until_expiration</
TimeoutInSec>
```

```

    <ExpiryDate ref="date_variable">expiration_date</ExpiryDate>
  </ExpirySettings>
  <Source>cache-response</Source>
</PopulateCache>

```

📌 Note

The ExpirySettings specifies when a cache entry should expire. When present, <TimeoutInSeconds> overrides both <TimeOfDay> and <ExpiryDate>.

Populate cache policy defines the following elements:

Element	Description
CacheKey	Configures a unique pointer to a piece of data stored in the cache. <ul style="list-style-type: none"> Prefix KeyFragment
CacheResource	Specifies the cache where messages should be stored. A default cache is available.
Scope	Enumeration used to construct a prefix for a cache key when a <Prefix> element is not provided in the <CacheKey> element.
ExpirySettings	Specifies when the cached value should expire. <ul style="list-style-type: none"> ExpiryDate TimeOfDay TimeoutInSec
Source	Specifies the variable whose value should be written to the cache.

Populate cache policy type defines the following error codes:

Error code	Cause
EntryCannotBeCached	An entry cannot be cached. The message object being cached is not an instance of a class that is Serializable.
InvalidCacheResourceReference	The cache specified in the <CacheResource> element does not exist.
CacheNotFound	The cache specified in the <CacheResource> element does not exist.

Following fault variables is set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	[prefix]: populatecache [policy_name]: The name of the policy to check.	populatecache.POP-CACHE-1.failed = true
fault.[error_name]	[error_name] = The specific error name to check for as listed in the table above.	fault.name Matches "EntryCannotBeCached"

Parent topic: [Caching Policies \[page 266\]](#)

Related Information

[Lookup Cache \[page 268\]](#)

[Invalidate Cache \[page 270\]](#)

[Response Cache \[page 272\]](#)

1.5.1.4.1.8.2 Lookup Cache

An OAuth access token is written to the cache using a Populate Cache policy. The OAuth token is retrieved for subsequent requests by a Lookup Cache policy.

At runtime, the LookupCache policy retrieves a value from the cache, assigning the value to the variable you specify with the AssignTo element (if no value is retrieved, the variable will not be set). It looks for the value based on a cache key created through configuration that combines the CacheKey and Scope elements. In other words, to retrieve a particular value-added to the cache by a PopulateCache policy, your LookupCache policy must have cache key-related elements configured in the same way as the PopulateCache policy.

You can retrieve cached values with the Lookup Cache policy.

You can attach this policy in the following locations:

ProxyEndpoint			TargetEndpoint			←Response	
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow		PostFlow
	•	•	•	•	•		•
	•	•	•	•	•		•
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

Code Syntax

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```

<LookupCache async="false" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
  <CacheKey>
    <KeyFragment ref="request.header.userid"></KeyFragment>
  </CacheKey>
  <Scope>Exclusive</Scope>
  <AssignTo>cache-response</AssignTo>
</LookupCache>

```

Element	Description
CacheKey	Configures a unique pointer to a piece of data stored in the cache.
CacheResource	Specifies the cache where messages should be stored. A default cache is available.
Scope	Enumeration used to construct a prefix for a cache key when a Prefix element is not provided in the CacheKey element. The list of supported values are: Global, Application, Proxy, Target, and Exclusive.
AssignTo	Specifies the variable where the cache entry is assigned after it has been retrieved from the cache.
Prefix	Specifies a value to use as a cache key prefix.
KeyFragment	Specifies a value that should be included in the cache key, creating a namespace for matching requests to cached responses.

The following predefined Flow variables are available after you customize the behavior of the cache you define in a Lookup Cache policy.

Flow Variables

Variables	Type	Permission	Description
lookupcache.{policy-name}.cachename	String	Read-Only	Returns the cache name used in the policy.
lookupcache.{policy-name}.cachekey	String	Read-Only	Returns the key used.
lookupcache.{policy-name}.cachehit	Boolean	Read-Only	True if the policy found a value for the specified cache key.
lookupcache.{policy-name}.assignto	String	Read-Only	Returns the variable to which cache is assigned.

Lookup Cache policy type defines the following error codes:

Error code	Occurs when
InvalidCacheResourceReference	The cache specified in the <CacheResource> element does not exist.
InvalidTimeout	The CacheLookupTimeoutInSeconds value must be greater than zero.
CacheNotFound	The cache specified in the <CacheResource> element does not exist.

Parent topic: [Caching Policies \[page 266\]](#)

Related Information

[Populate Cache \[page 266\]](#)

[Invalidate Cache \[page 270\]](#)

[Response Cache \[page 272\]](#)

1.5.1.4.1.8.3 Invalidate Cache

The cache can be invalidated explicitly by specifying an HTTP header. When a request that contains the specified HTTP header is received, the cache will be flushed.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

Code Syntax

```
<!-- The policy will clear the value of userId from the cache -->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InvalidateCache async="false" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
  <CacheKey>
    <KeyFragment ref="request.header.userid"></KeyFragment>
  </CacheKey>
  <Scope>Exclusive</Scope>
  <PurgeChildEntries>true</PurgeChildEntries>
</InvalidateCache>
```

Invalidate Cache policy defines the following attributes that are common to all policy parent elements:

Attribute	Description	Default	Presence
name	The internal name of the policy. Characters you can use in the name are restricted to: A-Z 0-9_ \- \$ %.	N/A	Required

Attribute	Description	Default	Presence
continueOnError	Set to false to return an error when a policy fails. This is expected behavior for most policies. Set to true to have flow execution continue even after a policy fails.	false	Optional
enabled	Set to true to enforce the policy. Set to false to "turn off" the policy. The policy will not be enforced even if it remains attached to a flow.	true	Optional
async	This attribute is deprecated.	false	Deprecated

Element	Description
CacheKey	Configures a unique pointer to a piece of data stored in the cache.
CacheResource	Specifies the cache where messages should be stored. A default cache is available.
Scope	Enumeration used to construct a prefix for a cache key when a <Prefix> element is not provided in the <CacheKey> element.
CacheContext	Specifies how to construct a cache key when a <Prefix> element value is not specified, or to clear cache entries added by another API proxy.
PurgeChildEntries	true to purge child cache entries when invalidating the cache. Default is false.
Prefix	Specifies a value to use as a cache key prefix.
KeyFragment	Specifies a value that should be included in the cache key, creating a namespace for matching requests to cached responses.

During the policy execution, the following errors can occur:

Error Name	Cause
InvalidCacheResourceReference	The cache specified in the <CacheResource> element does not exist.
CacheNotFound	The cache specified in the <CacheResource> element does not exist.

Parent topic: [Caching Policies \[page 266\]](#)

Related Information

[Populate Cache \[page 266\]](#)

[Lookup Cache \[page 268\]](#)

[Response Cache \[page 272\]](#)

1.5.1.4.1.8.4 Response Cache

This policy helps in caching data from a backend resource, thus reducing the number of requests to the resource. When applications make requests to the same URI, use this policy to return cached responses instead of forwarding all the requests to the backend server. This results in improving your API's performance through reduced latency and network traffic.

ResponseCache is useful in cases where the backend data used by your API is updated only periodically.

The maximum size for each cached object is 512 kb. You can configure the ResponseCache policy to include HTTP response headers in setting cache entry expiration and cache keys.

You can attach this policy in the following locations:

ProxyEndpoint			TargetEndpoint			←Response	
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow		PostFlow
	•	•	•	•	•		•
	•	•	•	•	•		•
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

Code Syntax

```
<ResponseCache async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
  <CacheKey><Prefix/>
    <KeyFragment ref="\request.uri\" type="\string\"/>
  </CacheKey>
  <Scope>Exclusive</Scope>
  <ExpirySettings><ExpiryDate/><TimeOfDay/>
    <TimeoutInSec ref="">60</TimeoutInSec>
  </ExpirySettings>
  <SkipCacheLookup>request.header.bypass-cache = \"true\"</
SkipCacheLookup>
  <SkipCachePopulation/>
</ResponseCache>
```

Elements & Attributes	Description
DisplayName (Optional)	<p>Label the policy with a different (from the name attribute), natural-language name. Use this in addition to the name attribute.</p> <p>If you omit this element, the value of the name attribute is used.</p> <p>Syntax: <DisplayName>Policy Display Name</DisplayName></p>
CacheLookupTimeoutInSeconds (Optional)	<p>This element indicates the number of seconds after which an unsuccessful cache search is considered as a missed cache.</p> <p>Syntax: <CacheLookupTimeoutInSeconds>60</CacheLookupTimeoutInSeconds></p>

Elements & Attributes	Description
CacheResource (Optional)	<p>This element indicates the cache where messages are stored. To use the included shared cache, skip this element.</p> <p>To administratively clear entries present in the cache, specify a CacheResource by name.</p> <p>Syntax: <code><CacheResource>my_cache_reserve</CacheResource></code></p>
ExcludeErrorResponse (Optional)	<p>The response cache policy currently caches both success and error HTTP responses by default.</p> <p>The default value is false. If you want to exclude caching target responses with HTTP error status codes, set this element to true, in which case only responses with status codes from 200 to 205 (success codes) are cached.</p> <p>Syntax: <code><ExcludeErrorResponse>>true</ExcludeErrorResponse></code></p>
SkipCacheLookup (Optional)	<p>This element (if the value evaluates to true) indicates that cache lookup should be skipped and the cache should be refreshed.</p> <p>Syntax: <code><SkipCacheLookup>variable-condition</SkipCacheLookup></code></p> <p>In the following example, the variable for bypass-cache is set to true, indicating that in an incoming header, the cache lookup is skipped and the cache is refreshed.</p> <p>Example: <code><SkipCacheLookup>request.header.bypass-cache = "true"</SkipCacheLookup></code></p>
SkipCachePopulation (Optional)	<p>This element (if the value evaluates to true) indicates that a write to the cache should be skipped.</p> <p>Syntax: <code><SkipCachePopulation>variable-condition</SkipCachePopulation></code></p> <p>In the following example, write to cache is skipped if the response status code is 200 or higher.</p> <p>Example: <code><SkipCachePopulation>response.status.code >= 200</SkipCachePopulation></code></p>

Elements & Attributes	Description
UseAcceptHeader (Optional)	<p>This element (if set to true) indicates that the response cache entry's cache key is appended with values from response accept headers.</p> <p>The following request headers are used while calculating the cache key:</p> <ul style="list-style-type: none"> • Accept • Accept-Language • Accept-Charset • Accept-Encoding <p>Syntax: <code><UseAcceptHeader>false</UseAcceptHeader></code></p>
UseResponseCacheHeader (Optional)	<p>This element (if set to true) indicates that HTTP response headers are considered while setting the time to live (TTL) of the response in the cache.</p> <p>While setting TTL, the values of the following response headers are considered, and compared with the values set by the ExpirySettings element:</p> <ul style="list-style-type: none"> • Cache-Control s-maxage • Cache-Control max-age • Expires <p>Syntax: <code><UseResponseCacheHeaders>false</UseResponseCacheHeaders></code></p>
Scope (Optional)	

Elements & Attributes	Description
CacheKey (Required)	<p data-bbox="544 344 1396 434">KeyFragment (Optional)</p> <p data-bbox="707 344 1396 434">The CacheKey element creates a unique pointer to a piece of data stored in cache, and has a size-limit of 2 KB each. It has two elements, KeyFragment and Prefix.</p> <p data-bbox="707 456 1396 546">The KeyFragment element indicates a value to be included in the cache key, in order to create a namespace for matching requests to cached responses.</p> <p data-bbox="707 568 1396 703">You can either provide a key (a static name) or a value (a dynamic variable) to the KeyFragment element. All the specified fragments combined (including the prefix) are concatenated to create the cache key.</p>

Sample Code

Syntax

```
<KeyFragment ref="variable_name" />
<KeyFragment>string</KeyFragment>
```

Example:

```
<KeyFragment>AccessToken</KeyFragment>
<KeyFragment ref="request_id" />
```

At runtime, the KeyFragment values are prepended with scope or prefix.

```
<CacheKey>
  <Prefix>User_Key</Prefix>
  <KeyFragment>AccessToken</KeyFragment>
  <KeyFragment ref="request_id" />
</CacheKey>
```

Elements & Attributes	Description
Prefix (Optional)	<p>The Prefix element indicates a string value that is used as a cache key prefix.</p> <p>Syntax: <code><Prefix>prefix_string</Prefix></code></p> <p>Use the prefix element along with the CacheKey and Scope elements (prefix overrides scope). If you want to specify your own value instead of a scope enumerated value, use prefix instead of scope.</p> <p>If you define a prefix, it prepends the cache key for entries written to the cache.</p> <div data-bbox="705 678 1396 969" style="background-color: #f0f0f0; padding: 10px;"> <p>Sample Code</p> <p>Example</p> <pre data-bbox="746 801 1361 936"><CacheKey> <Prefix>User_Key</Prefix> <KeyFragment>AccessToken</KeyFragment> <KeyFragment ref="request_id" /> </CacheKey></pre> </div>
ExpirySettings (Required)	<p>TimeoutInSec (Optional)</p> <p>This element indicates when a cache entry should expire. It includes the TimeoutInSec, TimeOfDay, and ExpiryDate elements.</p> <p>When present, TimeoutInSec overrides both TimeOfDay and ExpiryDate.</p> <p>The TimeoutInSec element is a variable with timeout value, which indicates the number of seconds after which a cache entry should expire.</p> <div data-bbox="705 1245 1396 1536" style="background-color: #f0f0f0; padding: 10px;"> <p>Sample Code</p> <p>Syntax</p> <pre data-bbox="746 1368 1361 1503"><ExpirySettings> <TimeoutInSec ref="duration_variable">seconds_to_expire< /TimeoutInSec> </ExpirySettings></pre> </div>

Elements & Attributes

Description

TimeOfDay

This element is a variable with the expiration time value (used in the format **hh.mm.ss**), which indicates the time of the day when a cache entry should expire.

The default time depends on the locale and timezone, which vary according to where the code is running.

↔ Sample Code

Syntax

```
<ExpirySettings>
  <TimeOfDay
    ref="time_variable">time_of_expiration
  </TimeOfDay>
</ExpirySettings>
```

ExpiryDate

This element is a variable (used in the format **mm-dd-yyyy**) which indicates the date on which a cache entry should expire.

↔ Sample Code

Syntax

```
<ExpirySettings>
  <ExpiryDate
    ref="{date_variable}">date_of_expiration</
  ExpiryDate>
</ExpirySettings>
```

Some predefined flow variables that are populated when a ResponseCache policy is executed are described in the below table:

Flow Variables

Variables	Type	Permission	Description
responsecache.{policy_name}.cachename	String	Read-Only	This variable returns the cache used in the policy.
	String	Read-Only	This variable returns the cache key used in the policy.
responsecache.{policy_name}.cachehit	String	Read-Only	True if the policy is executed successfully
responsecache.{policy_name}.invalidentry	String	Read-Only	True if the cache entry is invalid

Error messages that are seen when this policy triggers an error are described in the below table:responsecache.{policy_name}.cachekey

Error Messages

Error Name	Cause
InvalidTimeout	A negative number value was specified in the <CacheLookupTimeoutInSeconds> element.
InvalidCacheResourceReference	The name specified in the <CacheResource> element does not exist.
ResponseCacheStepAttachmentNotAllowedReq	The ResponseCache policy was attached more than once in the request path or to multiple request paths. For example, you cannot place it in both the Request PreFlow and PostFlow.
ResponseCacheStepAttachmentNotAllowedResp	The ResponseCache policy was attached to multiple response paths.
InvalidMessagePatternForErrorCode	The <SkipCacheLookup> or the <SkipCachePopulation> element in a ResponseCache policy contained an invalid condition.
CannotDeleteStepDefinition	You must detach the policy definition from the proxy flows before you can delete the policy.
CacheNotFound	The cache specified in the <CacheResource> element does not exist.

Parent topic: [Caching Policies \[page 266\]](#)

Related Information

[Populate Cache \[page 266\]](#)

[Lookup Cache \[page 268\]](#)

[Invalidate Cache \[page 270\]](#)

1.5.1.4.1.9 JavaScript

JavaScript Policy is used to configure the JavaScript code to execute within the context of an API proxy flow.

A JavaScript policy contains no actual code. Instead, a JavaScript policy references a JavaScript resource and defines the step in the API flow where the JavaScript executes. The JavaScript resource that is referenced by the JavaScript policy can be stored in the API Proxy bundle. The JavaScript resource always has a .js extension.

You can attach this policy in the following locations:

ProxyEndpoint			TargetEndpoint			←Response	
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow		PostFlow
	•	•	•	•	•		•
	•	•	•	•	•		•
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

Code Syntax

```
<!-- Use case: Remove forward slash from incoming URL present at the end
using a library function in a different script file -->
<Javascript async="false" continueOnError="false" enabled="true"
timeLimit="200" xmlns='http://www.sap.com/apimgmt'>
  <IncludeURL>jsc://lib.js</IncludeURL>
  <ResourceURL>jsc://maincode.js</ResourceURL>
</Javascript>
<!-- =====Content of lib.js ===== -->

//Trims the last chars from the string
function trimEnd(value, searchChar){
  if (value.charAt(value.length - 1) == searchChar) {
    value = value.substr(0, value.length - 1);
  }
  return value;
}
<!-- ===== Contents of maincode.js ===== -->
//Returns the proxy path suffix by escaping the trailing '/'
function getTrimmedPathSuffix(){
  return trimEnd(context.getVariable("proxy.pathsuffix"),'/');
}
context.setVariable("trimmedPathSuffix", getTrimmedPathSuffix());
```

Elements and Attributes	Description
timeLimit (required)	Specifies the maximum time (in milliseconds) that the script is permitted to execute.
ResourceURL (required)	Specifies the JavaScript resource (file). This is the main code file from which the execution begins. Syntax: <ResourceURL>jsc://example-javascript.js</ResourceURL>
IncludeURL (optional)	Specifies a JavaScript library to be loaded as dependency. Store libraries under /APIProxy/FileResources/<policy name>.js in your API proxy bundle. The scripts are evaluated in the order in which they are listed in the policy. Syntax: <IncludeURL>jsc://my-javascript-URL.js</IncludeURL>
Display name (optional)	Labels the policy in the management UI proxy editor with a different, natural-language name. If you skip this element, the value of the name attribute is applied to the policy. Syntax: <DisplayName>Policy-Display-Name</DisplayName>

During the policy execution, the following errors can occur:

Error Name	Cause
ScriptExecutionFailed	A runtime error occurred in the JavaScript code. See the fault string for details.
ScriptExecutionFailedLineNumber	An error occurred in the JavaScript code. See the fault string for details.
ScriptSecurityError	A security error occurred when the JavaScript executed. See the fault string for details.
WrongResourceType	In the <ResourceURL> and <IncludeURL> elements, you must refer to a JavaScript file correctly using the jsc resource type. For example, here is the correct way to refer to the JavaScript file in the policy: <ResourceURL>jsc://JavaScript-1.js</ResourceURL>
NoResourceForURL	The <ResourceURL> and <IncludeURL> elements refer to a JavaScript file that does not exist.
ScriptCompilationFailed	An error occurred during compilation of the JavaScript code. Refer to the error message for details.
InvalidResourceUrlFormat	This error occurs when the format of the resource URL specified within the <ResourceURL> or the <IncludeURL> element of the JavaScript policy is invalid, resulting in the deployment of the API proxy to fail.
InvalidResourceUrlReference	This error occurs when the <ResourceURL> or the <IncludeURL> elements refer to a JavaScript file that does not exist, resulting in the deployment of the API proxy to fail.

Following fault variables are set when the policy triggers an error at runtime:

Variable Set	Where	Example
[prefix].[policy_name].failed	The [prefix] is javascript. The [policy_name] is the name of the policy that threw the error.	javascript.JavaScript-1.failed = true
fault.[error_name]	[error_name] is the specific error name to check for as listed in the table above.	fault.name Matches "ScriptExecutionFailed"

Following is an example of an error response:

```

❏ Sample Code

{
  "fault": {
    "faultstring": "Execution of SetResponse failed with error:
Javascript runtime error: "ReferenceError: "status" is not defined.
(setresponse.js:6)\",
    "detail": {
      "errorcode": "steps.javascript.ScriptExecutionFailed"
    }
  }
}

```



```
}
```

Following is an example of a fault rule:

Sample Code

```
<FaultRule name="JavaScript Policy Faults">
  <Step>
    <Name>AM-CustomErrorResponse</Name>
    <Condition>(fault.name Matches "ScriptExecutionFailed") </Condition>
  </Step>
  <Condition>(javascript.JavaScript.failed = true) </Condition>
</FaultRule>
```

1.5.1.4.1.10 Java Script Object Model

This topic describes JavaScript model for API Management. JavaScript model enables you to use the JavaScript policy to add custom JavaScript to an API proxy.

API Management JavaScript object model defines objects that can be used in a JavaScript code executing within an API proxy flow. Objects defined by the JavaScript object model are within the scope of the API proxy flow. On executing the JavaScript, a scope is created and within the scope, the following object references are created: **context**, **request**, **response**, and **crypto**. You can also use a **print** function for debugging purpose.

Context Object

A context object is created for each request or response transaction executed by an API proxy. The context object exposes methods to get, set, and remove variables related to each transaction. Variables define properties specific to a transaction, and thus building logic that relies on these properties to execute custom behavior.

- **Scope:** Global; available throughout the API Proxy flow.
- **Child objects:** proxyRequest, proxyResponse, targetRequest, and targetResponse.
Child objects are scoped to the ambient request and response, either the proxy request and response or the target request and response. For example, if the JavaScript policy executes in the proxy endpoint part of the flow, then the context.proxyRequest and context.proxyResponse objects are in scope. If the JavaScript runs in a target flow, then the context.targetRequest and context.targetResponse objects are in scope.

Following table describes context objects and its children:

Context objects

Name	Description	Properties
context	A wrapper for the message processing pipeline context and the request and response Flows that are executed by the ProxyEndpoint and TargetEndpoint.	flow, session

Name	Description	Properties
context.proxyRequest	An object that represents the inbound request message to the ProxyEndpoint (from the requesting app to the API proxy)	headers, query parameters, method, body, url
context.targetRequest	An object that represents the outbound request message from the TargetEndpoint (from the API proxy to the back end service).	headers, query parameters, method, body, url
context.targetResponse	An object that represents the inbound target response message (from the backend service to the API proxy)	headers, content, status
context.proxyResponse	An object that represents the outbound proxy response message (from the API proxy to the requesting app)	headers, content, status
Context.flow	The name of the current flow.	
Context.session	A map of name/value pairs that you can use to pass objects between two different steps executing in the same context. For example: context.session['key'] = 123.	

context.*Request child objects

Each HTTP transaction executing in an API Proxy, creates two request message objects:

- Inbound: Request from client.
- Outbound: Request generated by API Proxy and submitted to the backend target.

Note

You can use the object `request` to access the properties in a request flow. The `request` object refers to either `context.proxyRequest` or `context.targetRequest`, depending on where in the flow your JavaScript code executes.

context.*Request child object properties

Property Name	Description
url	<p>The url property is a read/write convenience property that combines scheme, host, port, path, and query parameters for the targetRequest.</p> <p>The complete URL of the request is composed of the following properties:</p> <p>protocol: The protocol of the URL (for example, HTTP, HTTPS)</p> <p>port: The port (for example: 80, 443)</p> <p>host: The host of the URL (for example, www.example.com)</p> <p>path: The path of the URI (for example, /v1/mocktarget)</p> <p>When getting url, a URL is returned in the following format:</p> <pre>protocol://host:port/path?queryParams</pre> <p>Examples:</p> <pre>context.targetRequest.url = 'http://www.example.com/path?q1=1' context.targetRequest.protocol = 'https';</pre>
headers	<p>HTTP request headers as a mapping of String => List</p> <p>Examples:</p> <p>For this HTTP request:</p> <pre>POST /v1/blogs HTTP/1.1 Host: api.example.com Content-Type: application/json Authorization: Bearer y1SkZIJbdWybfs4fUQe9BqP0LH5Z</pre> <p>The following JavaScript:</p> <pre>context.proxyRequest.headers['Content-Type']; context.proxyRequest.headers['Authorization'];</pre> <p>returns the following values:</p> <pre>application/json Bearer y1SkZIJbdWybfs4fUQe9BqP0LH5Z</pre>
queryParams	<p>The request message query parameters as a mapping of String => List.</p> <p>Examples: "?city=PaloAlto&city=NewYork"</p> <p>can be accessed as:</p> <pre>context.proxyRequest.queryParams['city']; // == 'PaloAlto' context.proxyRequest.queryParams['city'][0] // == 'PaloAlto' context.proxyRequest.queryParams['city'][1]; // == 'NewYork' context.proxyRequest.queryParams['city'].length(); // == 2</pre>

Property Name	Description
method	<p>The HTTP verb (GET, POST, PUT, DELETE, PATCH, etc.) associated with the request</p> <p>Examples:</p> <p>For this request:</p> <pre>POST /v1/blogs HTTP/1.1 Host: api.example.com Content-Type: application/json Authorization: Bearer ylSkZIJbdWybfs4fUQe9BqP0LH5Z</pre> <p>The following JavaScript:</p> <pre>context.proxyRequest.method;</pre> <p>returns the following value</p> <p>POST</p>

Property Name	Description
body	<p>The message body (payload) of the HTTP request.</p> <p>The request body has the following members:</p> <pre>context.targetRequest.body.asXML;</pre> <pre>context.targetRequest.body.asJSON;</pre> <pre>context.targetRequest.body.asForm;</pre> <p>Examples:</p> <p>For an XML body:</p> <pre><customer number='1'> <name>Fred<name/> </customer/></pre> <p>To access the elements of the XML object as follows:</p> <pre>var name = context.targetRequest.body.asXML.name;</pre> <p>To access XML attributes, use the @ notation.</p> <pre>var number = context.targetRequest.body.asXML.@number;</pre> <p>For a JSON request body:</p> <pre>{ "a": 1, "b": "2" }</pre> <pre>var a = context.proxyRequest.body.asJSON.a; // == 1 var b = context.proxyRequest.body.asJSON.b; // == 2</pre> <p>To read form parameters:</p> <pre>"vehicle=Car&vehicle=Truck" v0 = context.proxyRequest.body.asForm['vehicle'][0]; v1 = context.proxyRequest.body.asForm['vehicle'][1];</pre>

context.*Response child objects

Each HTTP transaction executing in an API Proxy, creates two response message objects:

- Inbound: Response from the backend service.
- Outbound: Response sent to client.

📌 Note

You can use the object `response` to access the properties in a request flow. The response object refers to either `context.proxyResponse` or `context.targetResponse`, depending on where in the flow your JavaScript code executes.

context.*Response object properties

Property Name	Description
headers	The HTTP headers of the response message as a mapping of String => List. Example: <pre>var cookie = context.targetResponse.headers['Set-Cookie'];</pre>
status	The status code with status message as a property. Both status code and status message are available as properties. Example: <pre>var status = context.targetResponse.status.code; // 200 var msg = context.targetResponse.status.message; // "OK"</pre>
content	The HTTP body (payload content) of the response message. Response content has the following members: <pre>context.targetResponse.content.asXML; context.targetResponse.content.asJSON;</pre>

- **Context object methods**

Methods

Method Name	Description	Syntax
context.getVariable()	Retrieves the value of a predefined or custom variable.	<pre>context.getVariable("variable-name");</pre>
context.setVariable()	Sets the value for a custom variable or for any predefined variables.	<pre>context.setVariable("variable-name", value);</pre>
context.removeVariable()	Removes a variable from the context.	<pre>context.removeVariable('variable-name');</pre>

Request and Response objects

The request and response objects are "shorthand" references to the ambient request and response, either the proxy request and response or the target request and response. The objects these variables refer to depend upon the context in which the JavaScript policy executes. If the JavaScript runs in the flow of a proxy endpoint, then the request and response variables refer to context.proxyRequest and context.ProxyResponse. If the JavaScript runs in a target flow, then the variables refer to the context.targetRequest and context.targetResponse.

Crypto Object

Crypto object adds basic, high-performance cryptographic support to the JavaScript Object Model. Crypto object lets you perform basic cryptographic hashing functions in JavaScript.

- Scope: Global
- Hash objects supported by crypto:
 - **SHA-1:** You can create SHA-1 objects, update them, and convert them to hex and base64 values.
 - Create an SHA-1 object

- ```
var _sha1 = crypto.getSHA1();
```
- Update an SHA-1 object  
`_sha1.update(value);`
  - Return the SHA-1 object as a hex string  
`var _hashed_token = _sha1.digest();`
  - Return the SHA-1 object as a base64 string  
`var _hashed_token = _sha1.digest64();`
  - **SHA-256:** You can create SHA-256 objects, update them, and convert them to hex and base64 values.
    - Create an SHA-256 object  
`var _sha256 = crypto.getSHA256();`
    - Update an SHA-256 object  
`_sha256.update(value);`
    - Return the SHA-256 object as a hex string  
`var _hashed_token = _sha256.digest();`
    - Return the SHA-256 object as a base64 string  
`var _hashed_token = _sha256.digest64();`
  - **SHA-512 :** You can create SHA-512 objects, update them, and convert them to hex and base64 values.
    - Create an SHA-512 object  
`var _sha512 = crypto.getSHA512();`
    - Update an SHA-512 object  
`_sha512.update(value);`
    - Return the SHA-512 object as a hex string  
`var _hashed_token = _sha512.digest();`
    - Return the SHA-512 object as a base64 string  
`var _hashed_token = _sha512.digest64();`
  - **MD5:** You can create MD5 objects, update them, and convert them to hex and base64 values.
    - Create an MD5 object  
`var _md5 = crypto.getmd5();`
    - Update an MD5 object  
`_md5 .update(value);`
    - Return the MD5 object as a hex string  
`var _hashed_token = _md5.digest();`
    - Return the MD5 object as a base64 string  
`var _hashed_token = _md5.digest64();`
  - **Crypto date/time support:** The crypto object supports date/time formatting patterns. `crypto.dateFormat()` returns a date in the string format.  
Syntax:  
`crypto.dateFormat(format, [timezone], [time])`  
The following table shows the parameters and examples of Crypto date/time support:

| Parameter       | Description                                                                                                                                       | Example                                                                                                           |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|
| format (string) | The underlying implementation for this parameter is <code>java.text.SimpleDateFormat</code> , for example: <code>'YYYY-MM-DD HH:mm:ss.SSS'</code> | Get the current time, down to milliseconds: <code>var _now = crypto.dateFormat('YYYY-MM-DD HH:mm:ss.SSS');</code> |

| Parameter                   | Description                                                                          | Example                                                                                                                                                                                                            |
|-----------------------------|--------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| timezone (string, optional) | The underlying implementation for this parameter is java.util.TimeZone. Default: UTC | Get the current time for Pacific Time Zone:<br>var _pst = crypto.dateFormat('YYYY-MM-DD HH:mm:ss.SSS', 'PST');                                                                                                     |
| time (number, optional)     | A Unix timestamp value to format. Default: current time                              | Get the value of ten seconds from current time:<br>var _timeNow = Number(context.getVariable('system.timestamp'));<br>var ten_seconds = crypto.dateFormat('YYYY-MM-DD HH:mm:ss.SSS', 'PST', _timeNow + 10 * 1000); |

### Sample with Crypto

```
try {
 //get values to use with hash functions
 var salt = context.getVariable("salt") || 'SomeHardCodedSalt';
 var host = context.getVariable("request.header.Host");
 var unhashed_token = "";
 var _timeNow = Number(context.getVariable('system.timestamp'));
 var now = crypto.dateFormat('YYYY-MM-DD HH:mm:ss.SSS', 'PST', _timeNow);
 unhashed_token = "|" + now + "|" + host
 //generate a hash with the unhashedToken:
 var sha512 = crypto.getSHA512();
 sha512.update(salt);
 sha512.update(unhashed_token);
 //convert to base64
 var base64_token = sha512.digest64();
 // set headers
 context.setVariable("request.header.now", now);
 context.setVariable("request.header.token", base64_token);
} catch(e) {
 throw 'Error in Javascript';
}
```

### The print() function

If you are using the Java Script policy to execute custom Java Script code, then use the print() function to output debug information. Print function is directly available through Java Script Object Model. For example:

#### Sample Code

```
if (context.flow=="PROXY_REQ_FLOW") {
 print("In proxy request flow");
 var username = context.getVariable("request.queryparam.user");
 print("Got query param: " + username);
 context.setVariable("USER.name", username);
 print("Set query param: " + context.getVariable("USER.name"));
}
if (context.flow=="TARGET_REQ_FLOW") {
 print("In target request flow");
 var username = context.getVariable("USER.name");
 var url = "http://www.abc.com/user?";
 context.setVariable("target.url", url + "user=" + username);
 print("callout to URL: ", context.getVariable("target.url"));
```



```
}
```

## Using http Client

The http Client object is exposed to custom JavaScript code through the JavaScript object model. To attach custom JavaScript to an API proxy, you use the JavaScript policy. When the policy runs, the custom JavaScript code executes.

The httpClient object is useful for developing composite services or mashups. For example, you can consolidate multiple backend calls into a single API method. This object is commonly used as an alternative to the Service Callout policy.

Here's a basic usage pattern. Instantiate a Request object, assign to it a URL (e.g., to a backend service you want to call), and call httpClient.send with that request object.

### Sample Code

```
var myRequest = new Request();
myRequest.url = "http://www.abc.com";
var exchangeObj = httpClient.send(myRequest);
```

## httpClient Reference

HTTP Client exposes two methods: get() and send()

- httpClient.get()

Usage: `var exchangeObj = httpClient.get(url);`

Return: method returns an exchange object. This object has no properties, and it exposes the following methods:

- isError(): (boolean) Returns true if the httpClient was unable to connect to the server. HTTP status codes 4xx and 5xx result in isError() false, as the connection completed and a valid response code was returned. If isError() returns true, then a call to getResponse() returns the JavaScript undefined.
- isSuccess(): (boolean) Returns true if the send was complete and successful.
- isComplete(): (boolean) Returns true if the request is complete.
- waitForComplete(): Pauses the thread until the request is complete (by success or error).
- getResponse(): (object) Returns the response object if the httpClient.send() was complete and successful.
- getError(): (string) If the call to httpClient.send() resulted in an error, returns the error message as a string.

You can use the exchange object later to get the actual HTTP response, or to check whether the response has timed out. For example:

### Sample Code

```
var ex1 = httpClient.get("http://www.example.com?api1");
context.session["ex1"] = ex1; // Put the object into the session
var ex2 = httpClient.get("http://www.example.com?api2");
context.session["ex2"] = ex2; // Put the object into the session
```

- httpClient.send()

Usage: `var request = new Request(url, operation, headers); var exchangeObj = httpClient.send(request);`  
 Returns: A call to `httpClient.send()` returns an exchange object.

### 1.5.1.4.1.11 JSON to XML

API Management enables developers to convert messages from the JavaScript object notation (JSON) format to the extensible markup language (XML) format by using the JSON to XML policy type.

This policy is useful for enabling backend XML services to support RESTful apps that require JSON (for example, due to a lack of an XML parsing capabilities on the client).

In a typical mediation scenario, a JSON to XML policy on the inbound request flow is often paired with an XML to JSON policy on the outbound response flow. By combining policies this way, a JSON API can be exposed for services that natively support only XML.

For scenarios where APIs are consumed by diverse consumer applications which may require either JSON or XML, the format of the response can be dynamically set by configuring JSON to XML and XML to JSON policies to execute conditionally.

You can attach this policy in the following locations:

| ProxyEndpoint |          |      |          | TargetEndpoint |      |          |            |
|---------------|----------|------|----------|----------------|------|----------|------------|
| Request →     | PreFlow  | Flow | PostFlow | PreFlow        | Flow | PostFlow | ← Response |
|               | •        | •    | •        | •              | •    | •        |            |
|               | •        | •    | •        | •              | •    | •        |            |
|               | PostFlow | Flow | PreFlow  | PostFlow       | Flow | PreFlow  |            |

An example payload for the policy is as follows:

```

Code Syntax

<!-- The policy will convert the json response to corresponding xml response
for all elements where
the parent is rootElement and child is item inside it, if the root element of
json doent have name, the objectRootElementName
property defines the name in xml -->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<JSONToXML async="true" continueOnError="false" enabled="true" xmlns="http://
www.sap.com/apimgmt">
 <Options>
 <ArrayItemElementName>item</ArrayItemElementName>
 <ArrayRootElementName>rootelement</ArrayRootElementName>
 <ObjectRootElementName>objectroot</ObjectRootElementName>
 <InvalidCharsReplacement></InvalidCharsReplacement>
 <AttributePrefix></AttributePrefix>
 <AttributeBlockName></AttributeBlockName>
 <TextNodeName></TextNodeName>
 <NamespaceSeparator></NamespaceSeparator>
 <DefaultNamespaceNodeName></DefaultNamespaceNodeName>
 <NamespaceBlockName></NamespaceBlockName>
 <NullValue>I_AM_NULL</NullValue>
 </Options>
 <OutputVariable>response</OutputVariable>
 <Source>response</Source>

```

```
</JSONToXML>
```

Following table lists the elements and attributes that you can configure on this policy

Elements and Attributes	Description
Source (optional)	<p>The variable containing the JSON-formatted message to be converted to XML. Usually, you set this to request or response, depending on whether the message to be transformed is inbound or outbound.</p> <p>If you have not defined the source, then it is treated as message, resolving to a request when the policy is attached to a request flow, or a response when the policy is attached to a response flow.</p> <p>If the source variable cannot be resolved, or resolves to a non-message type, the policy throws an error.</p>
OutputVariable	<p>The variable name of the resultant XML-formatted message. Usually, you set this to request or response, depending on whether the message to be transformed is inbound or outbound. In most cases, a JSON request is transformed into an XML request. Similarly, a JSON response is usually transformed into an XML response.</p> <p>If OutputVariable is not specified, then the source variable is treated as OutputVariable. That is, the policy assumes that a JSON request, for example, is being converted into an XML request.</p> <div data-bbox="678 1086 1402 1227"><p><b>Note</b></p><p>This element is required when the variable defined in the Source element is a string.</p></div>
InvalidCharsReplacement	<p>To assist with handling invalid XML that may cause issues with a parser, this setting replaces any JSON elements that produce invalid XML with the string.</p> <p>For example, the following setting: <code>&lt;InvalidCharsReplacement&gt;_&lt;/InvalidCharsReplacement&gt;</code></p> <p>Converts this JSON object:</p> <pre>{   "First%%Name" : "Adam" }</pre> <p>to this XML structure: <code>&lt;First_Name&gt;Adam&lt;First_Name&gt;</code></p>

Elements and Attributes	Description
TextNodeName	<p data-bbox="659 344 1337 434">Converts a JSON property into an XML text node with the specified name. For example, the following setting: <code>&lt;TextNodeName&gt;age&lt;/TextNodeName&gt;</code></p> <p data-bbox="659 459 852 481">converts this JSON:</p> <pre data-bbox="659 506 1396 714"> {   "person": {     "firstName": "Adam",     "lastName": "Philip",     "age": 30   } } </pre> <p data-bbox="659 734 1355 799">to this XML structure:<code>&lt;person&gt;&lt;firstName&gt;Adam&lt;/firstName&gt;30&lt;lastName&gt;Philip&lt;/lastName&gt;&lt;/person&gt;</code></p> <p data-bbox="659 824 1374 889">If TextNodeName is not specified, the XML is generated, using the default setting for a text node:</p> <pre data-bbox="659 913 1396 1055"> &lt;person&gt;   &lt;firstName&gt;Adam&lt;/firstName&gt;   &lt;age&gt;30&lt;/age&gt;   &lt;lastName&gt;Philip&lt;/lastName&gt; &lt;/person&gt; </pre>
NullValue	<p data-bbox="659 1099 1150 1122">Indicates a null value. By default the value is NULL.</p> <p data-bbox="659 1146 1321 1211">For example the following setting: <code>&lt;NullValue&gt;NULL_VALUE&lt;/NullValue&gt;</code></p> <p data-bbox="659 1236 1374 1319">Converts the following JSON object: <code>{ "person" : "NULL_VALUE" }</code> to the following XML element: <code>&lt;person&gt;&lt;/person&gt;</code></p> <p data-bbox="659 1344 1398 1444">Where no value (or a value other than NULL_VALUE) is specified for the Null value, then the same payload converts to: <code>&lt;person&gt;NULL_VALUE&lt;/person&gt;</code></p>

## Elements and Attributes

## Description

### AttributeBlockName

Enables you to specify when JSON elements are converted into XML attributes (rather than XML elements).

For example, the following setting converts properties inside an object named #attrs into XML attributes:

```
<AttributeBlockName>#attrs</AttributeBlockName>
```

The following JSON object:

```
{
 "person" : {
 "#attrs" : {
 "firstName" : "Adam",
 "lastName" : "Philip"
 },
 "location" : "California"
 }
}
```

is converted to the following XML structure:

```
<person firstName="Adam"
lastName="Philip"><location>California</
location></person>
```

### AttributePrefix

Converts the property starting with the specified prefix into XML attributes.

Where the attribute prefix is set to #, for example:

```
<AttributePrefix>#</AttributePrefix>
```

Converts the following JSON object:

```
{
 "person" : {
 "#firstName" : "Adam",
 "#lastName" : "Philip",
 "location" : "California"
 }
}
```

to the following XML structure:

```
<person firstName="Adam"
lastName="Philip"><location>California</
location></person>
```

Elements and Attributes	Description
NamespaceBlockName	JSON has no support for namespaces, while XML documents often require them. The NamespaceBlockName enables you to define a JSON property that serves as the source of a namespace definition in the XML that is produced by the policy. (This means that the source JSON must provide a property that can be mapped into a namespace that is expected by application that consumes the resulting XML.)
DefaultNamespaceNodeName	
NameSpaceSeparator	

For example, the following settings:

```
<NamespaceBlockName>#namespaceblock</
NamespaceBlockName>
<DefaultNamespaceNodeName>$defaultname</
DefaultNamespaceNodeName>
<NamespaceSeparator>:</NamespaceSeparator>
```

indicates that a property called #namespaceblock exists in the source JSON that contains at least one namespace designated as the default. For example:

```
{
 "vehicle": {
 "#namespaceblock": {
 "$default": "http://www.w3.org/1999/
name",
 "xmlns:model": "http://www.w3.org/
1999/models"
 },
 "name": "Car",
 "models:name": "Alto"
 }
}
```

converts to:

```
<name xmlns="http://www.w3.org/1999/name"
xmlns:exp="http://www.w3.org/1999/models">
 <name>Car</name>
 <models:name>Alto</models:name>
</name>
```

Elements and Attributes	Description
<p>ArrayRootElementName</p> <p>ArrayItemElementName</p>	<p>Converts a JSON array into a list of XML elements with specified parent and child element names.</p> <p>For example, the following settings:</p> <pre data-bbox="659 472 1394 602">&lt;ArrayRootElementName&gt;Array&lt;/ ArrayRootElementName&gt; &lt;ArrayItemElementName&gt;Item&lt;/ ArrayItemElementName&gt;</pre> <p>Converts the following JSON array:</p> <pre data-bbox="659 674 1394 880">[   "Doctor" ,   {     "occupation": "Engineer"   },   "Scientist" ]</pre> <p>into the following XML structure:</p> <pre data-bbox="659 943 1394 1149">&lt;Array&gt;   &lt;Item&gt;Dcotor&lt;/Item&gt;   &lt;Item&gt;     &lt;occupation&gt;Engineer&lt;/occupation&gt;   &lt;/Item&gt;   &lt;Item&gt;Scientist&lt;/Item&gt; &lt;/Array&gt;</pre>
ObjectRootElementName	<p>Specifies the root element name when you convert from JSON, which does not have a named root element, to XML.</p> <p>For example, if the JSON appears as:</p> <pre data-bbox="659 1301 1394 1435">{   "xyz": "123",   "abc": "234" }</pre> <p>And you set the ObjectRootElementName as: &lt;ObjectRootElementName&gt;Root&lt;/ObjectRootElementName&gt;</p> <p>The resulting XML appears as:</p> <pre data-bbox="659 1626 1394 1760">&lt;Root&gt;   &lt;xyz&gt;123&lt;/xyz&gt;&gt;   &lt;abc&gt;234&lt;/abc&gt; &lt;/Root&gt;</pre>

During the policy execution, the following errors can occur:

## Error Cause

Error Name	Cause
SourceUnavailable	The variable specified in the Source element is not available or cannot be resolved.
ExecutionFailed	The input payload (JSON) is empty or the input (JSON) passed to XML policy is invalid or malformed.
OutputVariablesNotAvailable	The variable specified in the Source element of the JSON to XML Policy is of type string and the OutputVariable is not defined.
InCompatibleTypes	The type of the variable defined in the Source and Output-Variable element does not match. The valid types are message and string.
InvalidSourceType	The type of the variable used to define the Source element is invalid. The valid types are message and string.

The following fault variables are set when the policy triggers an error at runtime:

## Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	The variable [prefix] is jsontoxml.  The [policy_name] is the name of the policy that threw the error.	jsontoxml.JSON-to-XML-1.failed = true
fault.[error_name]	[error_name] = The specific error name to check for as listed in the table above.	fault.name Matches "SourceUnavailable"

Following is an example of an error response:

### Sample Code

```
{
 "fault": {
 "faultstring": "JSONToXML[JSON-to-XML-1]: Source abc is not available",
 "detail": {
 "errorcode": "steps.json2xml.SourceUnavailable"
 }
 }
}
```

Following is an example of a fault rule:

### Sample Code

```
<FaultRule name="JSON To XML Faults">
 <Step>
 <Name>AM-SourceUnavailableMessage</Name>
 <Condition>(fault.name Matches "SourceUnavailable") </Condition>
 </Step>
 <Step>
 <Name>AM-BadJSON</Name>
 <Condition>(fault.name = "ExecutionFailed")</Condition>
 </Step>
 <Condition>(jsontoxml.JSON-to-XML-1.failed = true) </Condition>
```



```
</FaultRule>
```

## Related Information

[XML to JSON \[page 387\]](#)

[XSL Transform \[page 399\]](#)

### 1.5.1.4.1.12 JSON Threat Protection

Minimizes the risk posed by content-level attacks by enabling specific limits on various JSON structures, such as arrays and strings.

You can attach this policy in the following locations:

Proxy Endpoint				Target Endpoint			
Request ➔	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	← Response
	●	●	●	●	●	●	
	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	

#### Note

JSON Threat Protection policy can be applied only to POST or PUT operations of an API. Applying this policy to a GET operation of an API results in an error.

An example payload for the policy is as follows:

#### Code Syntax

```
<JSONThreatProtection async="false" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <ArrayElementCount>15</ArrayElementCount>
 <ContainerDepth>15</ContainerDepth>
 <ObjectEntryCount>15</ObjectEntryCount>
 <ObjectEntryNameLength>25</ObjectEntryNameLength>
 <Source>request</Source>
 <StringValueLength>100</StringValueLength>
</JSONThreatProtection>
```

Elements and Attributes	Description
Array Element Count (Optional)	<p>This attribute specifies the maximum number of elements allowed in an array.</p> <p>The limit is not applied if you do not specify this element, or if you specify a negative integer.</p> <p>Syntax: <code>&lt;ArrayElementCount&gt;10&lt;/ArrayElementCount&gt;</code></p>
Container Depth (Optional)	<p>This attribute specifies the maximum container depth allowed for objects or arrays.</p> <p>For example, the container depth of an array containing an object, which contains another object is 3.</p> <p>The limit is not applied if you do not specify this element, or if you specify a negative integer.</p> <p>Syntax: <code>&lt;ContainerDepth&gt;5&lt;/ContainerDepth&gt;</code></p>
Object Entry Count (Optional)	<p>This attribute specifies the maximum number of entries allowed within an object.</p> <p>The limit is not applied if you do not specify this element, or if you specify a negative integer.</p> <p>Syntax: <code>&lt;ObjectEntryCount&gt;10&lt;/ObjectEntryCount&gt;</code></p>
Object Entry Length Name (Optional)	<p>This attribute specifies the maximum string length allowed for a property name within an object.</p> <p>The limit is not applied if you do not specify this element, or if you specify a negative integer.</p> <p>Syntax: <code>&lt;ObjectEntryNameLength&gt;100&lt;/ObjectEntryNameLength&gt;</code></p>
Source (Optional)	<p>This attribute indicates the message to be screened for JSON payload attacks.</p> <p><b>Request:</b> With a threat protection policy attached to any request flow, invalid messages return a 400 status code, along with a corresponding policy error message.</p> <p><b>Response:</b> Threat protection policy attached to any response flow, invalid messages still returns a 500 status code, and one of the corresponding policy error messages is thrown (rather than just ExecutionFailed).</p> <p>Syntax: <code>&lt;Source&gt;response&lt;/Source&gt;</code></p>

Elements and Attributes	Description
String Value Length (Optional)	<p>This attribute specifies the maximum length allowed for a string value.</p> <p>The limit is not applied if you do not specify this element, or if you specify a negative integer.</p> <p>Syntax: <code>&lt;StringValueLength&gt;200&lt;/StringValueLength&gt;</code></p>

During the policy execution, the following errors can occur:

Error Code

Error Name	HTTP Status	Cause
ExceededContainerDepth	500	JSONThreatProtection[policy_name]: Exceeded container depth at line [line_num]
ExceededObjectEntryCount	500	JSONThreatProtection[policy_name]: Exceeded object entry count at line [line_num]
ExceededArrayElementCount	500	JSONThreatProtection[policy_name]: Exceeded array element count at line [line_num]
ExceededObjectEntryNameLength	500	JSONThreatProtection[policy_name]: Exceeded object entry name length at line [line_num]
ExceededStringValueLength	500	JSONThreatProtection[policy_name]: Exceeded string value length at line [line_num]
Invalid JSON object	500	JSONThreatProtection[policy_name]: The input JSON Payload is invalid.
SourceUnavailable	500	<p>This error occurs when the message variable mentioned in the source element is either unavailable in the specific flow where the policy is being executed or it does not have a valid value (request, response, or message).</p> <p>JSONThreatProtection[policy_name]: Source [var_name] is not available</p>
NonMessageVariable	500	<p>This error occurs when the source element is set to a variable type which is not a message.</p> <p>JSONThreatProtection[policy_name]: Variable [var_name] does not resolve to a Message</p>
ExecutionFailed	500	JSONThreatProtection[policy_name]: Execution failed. reason: [string]

Following fault variables is set when the policy triggers an error at runtime:

## Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	[prefix]: jsonattack [policy_name]: The name of the policy to check.	jsonthreatprotection.JTP-SecureRequest.failed = true
fault.[error_name]	[error_name] = The specific error name to check for as listed in the table above.	fault.name Matches "SourceUnavailable"

Following is an example of an error response:

### Sample Code

```
{
 "fault": {
 "faultstring": "JSONThreatProtection[JTP-SecureRequest]: Execution failed. reason: JSONThreatProtection[JTP-SecureRequest]: Exceeded object entry name length at line 5",
 "detail": {
 "errorcode": "steps.jsonthreatprotection.ExecutionFailed"
 }
 }
}
```

Following is an example of a fault rule:

### Sample Code

```
<FaultRule name="JSON Threat Protection Policy Faults">
 <Step>
 <Name>CustomErrorResponse</Name>
 <Condition>(fault.name Matches "ExecutionFailed") </Condition>
 </Step>
 <Condition>(jsonattack.JTP-SecureRequest.failed = true) </Condition>
</FaultRule>
```

## Related Information

[JSON to XML \[page 290\]](#)

[XML Threat Protection \[page 401\]](#)

[Regular Expression Protection \[page 414\]](#)

### 1.5.1.4.1.13 JSON Web Tokens

This topic describes about JSON Web Tokens (JWT) policies available in API Management.

JSON Web Tokens, or JWT, are commonly used to share claims or assertions between connected applications. The JWT policies enable API proxies to:

- Generate signed JWTs.
- Verify digitally signed JWTs and claims within those JWTs.
- Decode signed JWTs without validating signatures on the token.

### Note

If this policy is not visible, then it is yet to be enabled for your data center and it will be enabled shortly.

### Algorithms used in JWT policy

- HMAC algorithm: The HMAC algorithm uses a secret key for creating and verifying signatures.
- RSA algorithm: The RSA algorithm uses a public/private key pair for the cryptographic signature. With RSA signatures, the signing party uses a private key to sign the JWT, and the verifying party uses the matching public key to verify the signature on the JWT.

### Parts of a JWT

A signed JWT encodes information in three parts:

- Header
- Payload
- Signature

Generate JWT policy creates all the parts, Verify JWT policy examines all the parts, and Decode JWT policy examines the header and payload.

### JSON Web Key Set (JWKS)

A JSON Web Key Set (JWKS) is a JSON structure that represents a set of JSON Web Keys. A JSON Web Key (JWK) is a JSON data structure that represents a cryptographic key.

JWKS structure: Each key element in the JWKS must include these attributes.

- kty - Must be set to RSA.
- kid (the key id) - Arbitrary value (no duplicates within a key set). If the inbound JWT bears a key ID, which is present in the set of JWKS, then the policy will use the correct public key to verify the JWT signature.
- n - The RSA key value "modulus".
- e - The RSA key value "exponent".

Following are optional elements and their required values:

- alg - If present, must be RS256.
- use - If present, must be sig.

The following JWKS includes the required elements and values

### Sample Code

```
{
 "keys": [
 {
 "kty": "RSA",
 "alg": "RS256",
 "use": "sig",
 "kid": "ca04df587b5a7cead80abee9ea8dcf7586a78e01",
 "n": "iXn-WmrwLLBa-QDiToBozpu4Y4ThKdwORWFXQa9I75pKOvPUjUjE2Bk05TUST7-
v7KDJcQ0_Nkd-
```

```

X9rMRV5LKgCa0_F8YgI30QS3bUm9orFryrdOc65PUIVfVxIwMZuGDYlhj6HEJVWIrOCZdcgNI1106B
asclckkUK40-
Eh7MaQrqb646ghFlG3zlgk9b2duHbDOq3s39ICPinRQWC6NqTYfqq7E8GN_NLY9srUcc_MswuUfMJ2
cKT6edrhLuIwIj_74YGkpOwilr2VswKsvJ7dcoiJxheKYvKDKtZfkbKrWETTJSGX2Xeh0DFB01qbKL
VvqkM2lFU2Qx1OgtTnrw" ,
 "e" : "AQAB"
},
{
 "kty" : "RSA" ,
 "alg" : "RS256" ,
 "use" : "sig" ,
 "kid" : "91412840c1019dedff1854b89938ad0a9078b871" ,

 "n" : "veQQDTKL8UKIRIuWxHLeRH0umTHtnm2LXe j56MungXuFZwmk_xccvsMpCtXmqhvEmHyAmKF06
YRRWAomHIwC7IBz9BsIjPYtOkvVkff_SCFoyuyDFkNsDsbTydIDUFAV5YlhZ0vgq1PJCzu8AfU9HoR
f0WIEnexpTDyWzlgNbg0b94Tlmw01C5kDRGDD33v8i-
RM4xXXyovBA_8R8D9t0aIzC9iVL418E0FOXAQ5xvrbvTXAkr17U4yIINUZ03q7i5tOxSA1z0s_-
CK6NN0LZcdQXddBFCCYXNmfhaiu-NeSvEPZMmDZGIFZihIvagPcs3pZfSC5ysyo3tu3r16tdw" ,
 "e" : "AQAB"
},
{
 "kty" : "RSA" ,
 "alg" : "RS256" ,
 "use" : "sig" ,
 "kid" : "f5010d8a5353b010c81fa45f1e43d2789b18bc9c" ,

 "n" : "2_jAH07j0ufDhv3sE2ky4m7w7xWbOZfMa1EI3MAOMdAcSkKGDlv67sN8OKTo3B8uDLIx8F81f
GToG4hUr8N08YXYFzqRpXk3b8_kzVDrFlZ7dSi_OhQXkXsnEdUPAn3CQu6a_ObQ7XAYrr2eF0L_unp
toQhanYte78LwOrPGKZTEKN6MEwHxz1yTR37yl88VNoV3WsLoeDxhomgtSD51iFGBFuJt5-f5x-
ZwXdDKgNgdIA7k8YYw246o47OKrb9xGR62qi0Mahxot_523BLyY18CUgbbp-
VBPPVyseNOrpUQarEFcAi3Pab4vwAzZoA8NVyv17aF2QRdIMIXoDmPw" ,
 "e" : "AQAB"
}
]
}

```

## Related Information

[Generate JWT \[page 302\]](#)

[Verify JWT \[page 311\]](#)

[Decode JWT \[page 319\]](#)

### 1.5.1.4.1.13.1 Generate JWT

This topic describes about Generate JSON Web Token(JWT) Policy.

This policy generates a signed JWT, with a configurable set of claims. The JWT can then be returned to clients, transmitted to backend targets, or used in other ways.

You can attach this policy in the following locations:

ProxyEndpoint			TargetEndpoint				
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	←Response
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

### Generate a JWT signed with the HS256 algorithm

The sample provided below generates a new JWT and signs it using the HS256 algorithm. HS256 relies on a shared secret for both signing and verifying the signature.

#### Code Syntax

```
<!-- Generate JWT policy -->
<GenerateJWT async=\"false\" continueOnError=\"false\" enabled=\"true\"
xmlns=\"http://www.sap.com/apimgmt\">
 <Algorithm>HS256</Algorithm>
 <IgnoreUnresolvedVariables>>false</IgnoreUnresolvedVariables>
 <SecretKey>
 <Value ref=\"private.secretkey\"/>
 <Id>1918290</Id>
 </SecretKey>
 <ExpiresIn>1h</ExpiresIn>
 <Subject>subject-subject</Subject>
 <Issuer>urn://apim-JWT-policy-test</Issuer>
 <Audience>audience1,audience2</Audience>
 <Id/>
 <AdditionalClaims>
 <Claim name=\"additional-claim-name\" type=\"string\">additional-
claim-value-goes-here</Claim>
 </AdditionalClaims>
 <OutputVariable>jwt-variable</OutputVariable>
</GenerateJWT>
```

The resulting JWT will have this header and payload:

#### Sample Code

```
header
{
 "typ" : "JWT",
 "alg" : "HS256",
}
payload
{
 "sub" : "subject-subject",
 "iss" : "urn://apim-JWT-policy-test",
 "aud" : "additional-claim-name",
 "iat" : 1506553019,
 "exp" : 1506556619,
 "jti" : "BD1FF263-3D25-4593-A685-5EC1326E1F37",
 "additional-claim-name": "additional-claim-value-goes-here"
}
```

## Generate a JWT signed with the RS256 algorithm.

This example policy generates a new JWT and signs it using the RS256 algorithm. Generating an RS256 signature relies on an RSA private key, which must be provided in PEM-encoded form. When this policy action is triggered, Edge encodes and digitally signs the JWT, including the claims. To learn about the parts of a JWT and how they're encrypted and signed, refer to RFC7519.

### Code Syntax

```
<!-- The policy -->
<GenerateJWT async=\"false\" continueOnError=\"false\" enabled=\"true\"
xmlns=\"http://www.sap.com/apimgmt\">
 <Algorithm>RS256</Algorithm>
 <IgnoreUnresolvedVariables>>false</IgnoreUnresolvedVariables>
 <PrivateKey>
 <Value ref=\"private.privatekey\"/>
 <Id ref=\"private.privatekey-id\"/>
 <Password ref=\"private.privatekey-password\"/>
 </PrivateKey>
 <ExpiresIn>1h</ExpiresIn>
 <Subject>subject-subject</Subject>
 <Issuer>urn://apim-JWT-policy-test</Issuer>
 <Audience>audience1,audience2</Audience>
 <Id/>
 <AdditionalClaims>
 <Claim name=\"additional-claim-name\" type=\"string\">additional-
claim-value-goes-here</Claim>
 </AdditionalClaims>
 <OutputVariable>jwt-variable</OutputVariable>
</GenerateJWT>
```

Following table lists the elements and attributes that you can configure on this policy.

Elements and Attributes	Description
Algorithm	<p>Specifies the encryption algorithm to sign the token. HS256 employs a shared secret. RS256 employs a public/secret key pair.</p> <p>Supported value: HS256, HS384, HS512, RS256, RS384, RS512</p> <pre>&lt;Algorithm&gt;HS256 RS256&lt;/Algorithm&gt;</pre>
Audience (optional)	<p>The policy generates a JWT containing an aud claim set to the specified value. This claim identifies the recipients that the JWT is intended for.</p> <pre>&lt;Audience&gt;audience&lt;/Audience&gt;</pre>



## Elements and Attributes

## Description

AdditionalClaims (optional)

Specify additional claims in the payload of the JWT. You can specify the claim explicitly as string, a number, a boolean, a map, or an array. A map is simply a set of name/value pairs.

```
<AdditionalClaims>
 <Claim name='claim1'>explicit-value-of-
claim-here</Claim>
 <Claim name='claim2' ref='var-name' />
 <Claim name='claim3' ref='var-name'
type='boolean' />
</AdditionalClaims>
```

The <Claim> element takes these attributes:

- name: name of the claim.

### Note

Don't use any of the registered claim names in this element. They include: "iss", "sub", "aud", "iat", "exp", "nbf", and "jti".

- ref: The name of a flow variable. If present, the policy will use the value of this variable as the claim. If both a ref attribute and an explicit claim value are specified, the explicit value is the default, and is used if the referenced flow variable is unresolved.
- type: One of: string (default), number, boolean, or map
- array: Set to true to indicate if the value is an array of types. Default: false.

Only name attribute is mandatory.

Elements and Attributes	Description
AdditionalHeaders (optional)	<p data-bbox="659 344 1145 367">Specify additional claim in the header for the JWT.</p> <pre data-bbox="671 394 1396 647"> &lt;AdditionalHeaders&gt;   &lt;Claim name='claim1'&gt;explicit-value-of-claim-here&lt;/Claim&gt;   &lt;Claim name='claim2' ref='var-name' /&gt;   &lt;Claim name='claim3' ref='var-name' type='boolean' /&gt;   &lt;Claim name='claim4' ref='var-name' type='string' array='true' /&gt; &lt;/AdditionalHeaders&gt; </pre> <p data-bbox="659 669 1094 692">The &lt;Claim&gt; element takes these attributes:</p> <ul data-bbox="671 719 951 741" style="list-style-type: none"> <li>• name: name of the claim.</li> </ul> <div data-bbox="707 768 1396 920" style="border: 1px solid #ccc; padding: 5px; margin: 10px 0;"> <p data-bbox="730 779 826 801"><b>Note</b></p> <p data-bbox="730 835 1369 898">Don't use any of the registered claim names in this element. They include: "iss", "sub", "aud", "iat", "exp", "nbf", and "jti".</p> </div> <ul data-bbox="671 931 1385 1178" style="list-style-type: none"> <li>• ref: The name of a flow variable. If present, the policy uses the value of this variable as the claim. If both a ref attribute and an explicit claim value are specified, the explicit value is the default, and is used if the referenced flow variable is unresolved.</li> <li>• type: One of: string (default), number, boolean, or map</li> <li>• array: Set to true to indicate if the value is an array of types. Default: false.</li> </ul> <p data-bbox="659 1200 992 1223">Only name attribute is mandatory.</p>
ExpiresIn (Optional)	<p data-bbox="659 1249 1329 1272">Specifies the lifespan of the JWT in seconds, minutes, hours, or days.</p> <p data-bbox="659 1301 1129 1323">Supported value: Time units can be specified as:</p> <ul data-bbox="671 1350 911 1503" style="list-style-type: none"> <li>• s = seconds (default)</li> <li>• m = minutes</li> <li>• h = hours</li> <li>• d = days</li> </ul> <p data-bbox="659 1525 1390 1588">For example, an ExpiresIn = 10d is equivalent to an ExpiresIn of 864000s or 864000s .</p> <pre data-bbox="671 1603 1396 1659"> &lt;ExpiresIn&gt;time-value&lt;/ExpiresIn&gt; </pre>
Id (Optional)	<p data-bbox="659 1682 1377 1771">The JWT ID (jti) claim is a unique identifier for the JWT. Id attribute generates a JWT with the specific jti claim. When the text value and ref attribute are both empty, the policy generates a jti containing a random UUID.</p> <pre data-bbox="671 1794 1396 1957"> &lt;Id&gt;explicit-jti&lt;/Id&gt; -or- &lt;Id ref='varname' /&gt; -or- &lt;Id/&gt; </pre>

Elements and Attributes	Description															
IgnoreUnresolvedVariables (Optional)	<p>Set to false if you want the policy to throw an error when any referenced variable specified in the policy is unresolvable. Set to true to treat any unresolvable variable as an empty string</p> <pre>&lt;IgnoreUnresolvedVariables&gt;true false&lt;/IgnoreUnresolvedVariables&gt;</pre>															
Issuer (Optional)	<p>The policy generates a JWT containing a claim with name iss, with a value set to the specified value. A claim that identifies the issuer of the JWT.</p> <pre>&lt;Issuer ref='variable-name-here' /&gt; &lt;Issuer&gt;issuer-string-here&lt;/Issuer&gt;</pre>															
NotBefore (Optional)	<p>Specifies an absolute time value for the token's expiration. The token is not valid until the specified time.</p> <pre>&lt;NotBefore&gt;2019-06-18T11:00:11-07:00&lt;/NotBefore&gt;</pre> <p>Supported values:</p> <table border="1"> <thead> <tr> <th>Name</th> <th>Format</th> <th>Example</th> </tr> </thead> <tbody> <tr> <td>sortable</td> <td>yyyy-MM-dd'T'HH:mm:ss.SSSZ</td> <td>2019-06-18T11:00:11.269-0700</td> </tr> <tr> <td>RFC 1123</td> <td>EEE, dd MMM yyyy HH:mm:ss zzz</td> <td>Tue, 18 Jun 2019 11:00:21 PDT</td> </tr> <tr> <td>RFC 850</td> <td>EEEE, dd-MMM-yy HH:mm:ss zzz</td> <td>Tuesday, 18 Jun 2019 11:00:21 PDT</td> </tr> <tr> <td>ANCI-C</td> <td>EEE MMM d HH:mm:ss yyyy</td> <td>Tue Jun 18 11:00:21 2017</td> </tr> </tbody> </table>	Name	Format	Example	sortable	yyyy-MM-dd'T'HH:mm:ss.SSSZ	2019-06-18T11:00:11.269-0700	RFC 1123	EEE, dd MMM yyyy HH:mm:ss zzz	Tue, 18 Jun 2019 11:00:21 PDT	RFC 850	EEEE, dd-MMM-yy HH:mm:ss zzz	Tuesday, 18 Jun 2019 11:00:21 PDT	ANCI-C	EEE MMM d HH:mm:ss yyyy	Tue Jun 18 11:00:21 2017
Name	Format	Example														
sortable	yyyy-MM-dd'T'HH:mm:ss.SSSZ	2019-06-18T11:00:11.269-0700														
RFC 1123	EEE, dd MMM yyyy HH:mm:ss zzz	Tue, 18 Jun 2019 11:00:21 PDT														
RFC 850	EEEE, dd-MMM-yy HH:mm:ss zzz	Tuesday, 18 Jun 2019 11:00:21 PDT														
ANCI-C	EEE MMM d HH:mm:ss yyyy	Tue Jun 18 11:00:21 2017														
OutputVariable (Optional)	<p>JWT generated by this policy is placed in the variable specified in this attribute. By default it is placed into the flow variable message.content.</p> <pre>&lt;OutputVariable&gt;jwt-var&lt;/OutputVariable&gt;</pre>															
<PrivateKey/Id> (Optional)	<p>Specifies the key ID (kid) to include in the JWT header.</p> <pre>&lt;PrivateKey&gt;   &lt;Id ref="flow-variable-name-here" /&gt; &lt;/PrivateKey&gt; or &lt;PrivateKey&gt;   &lt;Id&gt;your-id-value-here&lt;/Id&gt; &lt;/PrivateKey&gt;</pre>															

Elements and Attributes	Description
<PrivateKey/Password> (Optional)	<p>Password to decrypt the private key, if necessary. Use the ref attribute to pass the key in a flow variable. Use this element only when the algorithm is an RSA variant.</p> <pre>&lt;PrivateKey&gt;   &lt;Password ref="private.privatekey-password" /&gt; &lt;/PrivateKey&gt;</pre>
<PrivateKey/Value> (Optional)	<p>Specifies a PEM-encoded private key used to sign the JWT. Use the ref attribute to pass the key in a flow variable. Use this only with the GenerateJWT policy, when the algorithm is an RSA variant.</p> <pre>&lt;PrivateKey&gt;   &lt;Value ref="private.variable-name-here" /&gt; &lt;/PrivateKey&gt;</pre>
<SecretKey/Id> (Optional)	<p>Specifies the key ID (kid) to include in the JWT header of a JWT signed with an HMAC algorithm.</p> <pre>&lt;SecretKey&gt;   &lt;Id ref="flow-variable-name-here" /&gt; &lt;/SecretKey&gt; or &lt;SecretKey&gt;   &lt;Id&gt;your-id-value-here&lt;/Id&gt; &lt;/SecretKey&gt;</pre>
<SecretKey/Value> (Optional)	<p>Provides the secret key used to verify or sign tokens with an HMAC algorithm. Use only when the algorithm is one of HS256, HS384, HS512. Use the ref attribute to pass the key in a flow variable.</p> <pre>&lt;SecretKey&gt;   &lt;Value ref="private.your-variable-name" /&gt; &lt;/SecretKey&gt;</pre>
<Subject> (Optional)	<p>The policy generates a JWT containing a sub claim, set to the specified value. This claim identifies or makes a statement about the subject of the JWT.</p> <pre>&lt;Subject&gt;subject-string-here&lt;/Subject&gt; -or - &lt;Subject ref="flow_variable" /&gt; - For example - &lt;Subject ref="sap.developer.email" /&gt;</pre>

The following flow variables are available after the policy is executed:

#### Flow Variables

Variable	Description
header.algorithm	Signing algorithm used on JWT.
claim.subject	JWT subject claim.
claim.issuer	JWT issuer claim.

Variable	Description
claim.audience	JWT audience claim. This value may be a string, or an array of strings.
claim.expiry	Expiration date or time, expressed in seconds since epoch.
expiry_formatted	Expiration date or time, formatted as a human readable string. Example: 2019-18-28T21:30:45.000+0000
seconds_remaining	Number of seconds before the token expires. If the token is expired, this number will be negative.
time_remaining_formatted	Time remaining before the token expires, formatted as a human-readable string. Example: 00:59:59.926
is_expired	true or false
claim.issuedat	Token issued Date, expressed in seconds since epoch.
claim.notbefore	If the JWT includes a nbf claim, this variable will contain the value. This is expressed in seconds since epoch.
valid	In the case of VerifyJWT, this variable will be true when the signature is verified, and the current time is before the token expiry, and after the token notBefore value, if they are present. Otherwise false.  In the case of DecodeJWT, this variable is not set.
claim.name	The value of the named claim (standard or additional) in the payload. One of these will be set for every claim in the payload.
header.name	The value of the named header (standard or additional). One of these will be set for every additional header in the header portion of the JWT.
header.kid	The Key ID, if added when the JWT was generated.
header.type	Will be set to JWT.
payload-claim-names	An array of claims supported by the JWT.
payload-json	Payload in JSON format.
header-json	Header in JSON format.

During the policy execution, the following errors can occur:

Error Cause

Error Name	Cause
steps.jwt.AlgorithmInTokenNotPresentInConfiguration	Occurs when the verification policy has multiple algorithms.
steps.jwt.AlgorithmMismatch	The algorithm specified in the Generate policy didn't match the one expected in the Verify policy. The algorithms specified must match.
steps.jwt.FailedToDecode	The policy was unable to decode the JWT. The JWT is possibly corrupted.
steps.jwt.GenerationFailed	The policy was unable to generate the JWT.
steps.jwt.InsufficientKeyLength	For a key less than 32 bytes for the HS256 algorithm, less than 48 bytes for the HS386 algorithm, and less than 64 bytes for the HS512 algorithm.

Error Name	Cause
steps.jwt.InvalidClaim	For a missing claim or claim mismatch, or a missing header or header mismatch.
steps.jwt.InvalidCurve	The curve specified by the key is not valid for the Elliptic Curve algorithm.
steps.jwt.InvalidJsonFormat	Invalid JSON found in the header or payload.
steps.jwt.InvalidToken	This error occurs when the JWT signature verification fails.
steps.jwt.JwtAudienceMismatch	The audience claim failed on token verification.
steps.jwt.JwtIssuerMismatch	The issuer claim failed on token verification.
steps.jwt.JwtSubjectMismatch	The subject claim failed on token verification.
steps.jwt.KeyIdMissing	The Verify policy uses a JWKS as a source for public keys, but the signed JWT does not include a kid property in the header
steps.jwt.KeyParsingFailed	The public key could not be parsed from the given key information.
steps.jwt.NoAlgorithmFoundInHeader	Occurs when the JWT contains no algorithm header.
steps.jwt.NoMatchingPublicKey	The Verify policy uses a JWKS as a source for public keys, but the kid in the signed JWT is not listed in the JWKS.
steps.jwt.SigningFailed	In GenerateJWT, for a key less than the minimum size for the HS384 or HS512 algorithms
steps.jwt.TokenExpired	The policy attempts to verify an expired token.
steps.jwt.TokenNotYetValid	The token isn't yet valid.
steps.jwt.UnknownException	An unknown exception occurred.
steps.jwt.WrongKeyType	Wrong type of key specified. For example, if you specify an RSA key for an Elliptic Curve algorithm, or a curve key for an RSA algorithm.

The following fault variables are set when the policy triggers an error at runtime:

#### Fault Variables

Variable Set	Where	Example
fault.name="fault_name"	fault_name is the name of the fault, as listed in the Runtime errors table above. The fault name is the last part of the fault code.	fault.name Matches "TokenExpired"
jwt.policy_name.failed	policy_name is the user-specified name of the policy that threw the fault.	jwt.JWT-Policy.failed = true

Following is an example of a fault rule:

#### Sample Code

```
<FaultRules>
 <FaultRule name="JWT Policy Errors">
 <Step>
 <Name>JavaScript-1</Name>
```

```

 <Condition>(fault.name Matches "TokenExpired")</Condition>
 </Step>
 <Condition>jwt.JWT-1.failed=true</Condition>
</FaultRule>
</FaultRules>

```

## Related Information

[JSON Web Tokens \[page 300\]](#)

[Verify JWT \[page 311\]](#)

[Decode JWT \[page 319\]](#)

### 1.5.1.4.1.13.2 Verify JWT

This policy describes about Verify JSON Web Token(JWT) Policy.

This policy verifies a signed JWT, with a configurable set of claims. When this policy executes, API Management verifies the signature of a JWT, and verifies that the JWT is valid according to the expiry and not-before times if they're present. The policy can optionally also verify the values of specific claims on the JWT, such as the subject, the issuer, the audience, or the value of additional claims. If the JWT is verified and valid, then all of the claims contained within the JWT are extracted into context variables for use by subsequent policies or conditions, and the request is allowed to proceed. If the JWT signature can't be verified or if the JWT is invalid because of one of the timestamps, all processing stops and an error is returned in the response.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

#### Verify a JWT signed with the HS256 algorithm

The sample provided below verifies a JWT signed with the HS256 encryption algorithm, HMAC using a SHA-256 checksum. HS256 relies on a shared secret for both signing and verifying the signature.

#### Code Syntax

```

<VerifyJWT async="\false\" continueOnError="\false\" enabled="\true\"
xmlns="\http://www.sap.com/apimgmt\">
 <Algorithm>HS256</Algorithm>
 <Source>request.formparam.jwt</Source>
 <IgnoreUnresolvedVariables>>false</IgnoreUnresolvedVariables>
 <SecretKey>
 <Value ref="private.secretkey"/>
 </SecretKey>

```

```

<Subject>subject-subject</Subject>
<Issuer>urn://apim-JWT-policy-test</Issuer>
<Audience>audiencel,audience2</Audience>
<AdditionalClaims>
 <Claim name=\"additional-claim-name\" type=\"string\">additional-
claim-value-goes-here</Claim>
</AdditionalClaims>
</VerifyJWT>

```

Verify a JWT signed with the RS256 algorithm

This example policy verifies a JWT that was signed using the RS256 algorithm. For signing, a private key must be provided, and to verify, you need to provide the corresponding public key.

### Code Syntax

```

<VerifyJWT async=\"false\" continueOnError=\"false\" enabled=\"true\"
xmlns=\"http://www.sap.com/apimgmt\">
 <Algorithm>RS256</Algorithm>
 <Source>request.formparam.jwt</Source>
 <IgnoreUnresolvedVariables>>false</IgnoreUnresolvedVariables>
 <PublicKey>
 <Value ref=\"public.publickey\"/>
 </PublicKey>
 <Subject>subject-subject</Subject>
 <Issuer>urn://apim-JWT-policy-test</Issuer>
 <Audience>audiencel,audience2</Audience>
 <AdditionalClaims>
 <Claim name=\"additional-claim-name\" type=\"string\">additional-
claim-value-goes-here</Claim>
 </AdditionalClaims>
</VerifyJWT>

```

The resulting JWT will have this header and payload and is valid, if the signature can be verified with the provided public key.

### Sample Code

```

header
{
 "typ" : "JWT",
 "alg" : "RS256"
}
payload
{
 "sub" : "subject-subject",
 "iss" : "urn://apim-edge-JWT-policy-test",
 "aud" : "audiencel,audience2",
 "additional-claim-name": "additional-claim-value-goes-here"
}

```

However, a JWT with the same header but different payload as shown below is invalid, even if the signature is verified. The "sub" claim included in the JWT does not match the required value of the "Subject" element as specified in the policy configuration.

### Sample Code

```

{

```



```

"sub" : "subject1",
"iss" : "urn://apim-edge-JWT-policy-test",
"aud" : "audience1,audience2",
"additional-claim-name": "additional-claim-value-goes-here"
}

```

Following table lists the elements and attributes that you can configure on this policy

Elements and Attributes	Description
Algorithm	<p>Specifies the encryption algorithm to sign the token. HS256 employs a shared secret. RS256 employs a public/secret key pair.</p> <p>Supported value: HS256, HS384, HS512, RS256, RS384, RS512</p> <pre>&lt;Algorithm&gt;HS256 RS256&lt;/Algorithm&gt;</pre>
Audience (optional)	<p>The policy verifies that the audience claim in the JWT matches the value specified in the configuration. If there is no match, the policy throws an error.</p> <pre>&lt;Audience&gt;audience&lt;/Audience&gt;</pre>
AdditionalClaims (optional)	<p>Validates additional claims in the payload of the JWT. An additional claim can be a string, a number, a boolean, a map, or an array. A map is simply a set of name/value pairs.</p> <pre> &lt;AdditionalClaims&gt;   &lt;Claim name='claim1'&gt;explicit-value-of-claim-here&lt;/Claim&gt;   &lt;Claim name='claim2' ref='var-name' /&gt;   &lt;Claim name='claim3' ref='var-name' type='boolean' /&gt; &lt;/AdditionalClaims&gt; </pre> <p>The &lt;Claim&gt; element takes these attributes:</p> <ul style="list-style-type: none"> <li>name: name of the claim.</li> </ul> <div data-bbox="724 1406 1401 1556" data-label="Text"> <p><b>Note</b></p> <p>Don't use any of the registered claim names in this element. They include: "iss", "sub", "aud", "iat", "exp", "nbf", and "jti".</p> </div> <ul style="list-style-type: none"> <li>ref: The name of a flow variable. If present, the policy will use the value of this variable as the claim. If both a ref attribute and an explicit claim value are specified, the explicit value is the default, and is used if the referenced flow variable is unresolved.</li> <li>type: One of: string (default), number, boolean, or map</li> <li>array: Set to true to indicate if the value is an array of types. Default: false.</li> </ul> <p>Only name attribute is mandatory.</p>

Elements and Attributes	Description
AdditionalHeaders (optional)	<p data-bbox="659 344 1161 367">Validates additional claim in the header for the JWT.</p> <pre data-bbox="659 389 1396 645"> &lt;AdditionalHeaders&gt;   &lt;Claim name='claim1'&gt;explicit-value-of-claim-here&lt;/Claim&gt;   &lt;Claim name='claim2' ref='var-name' /&gt;   &lt;Claim name='claim3' ref='var-name' type='boolean' /&gt;   &lt;Claim name='claim4' ref='var-name' type='string' array='true' /&gt; &lt;/AdditionalHeaders&gt; </pre> <p data-bbox="659 667 1094 689">The &lt;Claim&gt; element takes these attributes:</p> <ul data-bbox="659 712 951 741" style="list-style-type: none"> <li>• name: name of the claim.</li> </ul> <div data-bbox="703 763 1396 920" style="border: 1px solid #ccc; padding: 5px; margin: 10px 0;"> <p data-bbox="727 779 826 808"><b>Note</b></p> <p data-bbox="727 831 1382 898">Do not use any of the registered claim names in this element. They include: "iss", "sub", "aud", "iat", "exp", "nbf", and "jti".</p> </div> <ul data-bbox="659 931 1386 1178" style="list-style-type: none"> <li>• ref: The name of a flow variable. If present, the policy uses the value of this variable as the claim. If both a ref attribute and an explicit claim value are specified, the explicit value is the default, and is used if the referenced flow variable is unresolved.</li> <li>• type: One of: string (default), number, boolean, or map</li> <li>• array: Set to true to indicate if the value is an array of types. Default: false.</li> </ul> <p data-bbox="659 1200 994 1223">Only name attribute is mandatory.</p>
Id (Optional)	<p data-bbox="659 1249 1390 1339">The JWT ID (jti) claim is a unique identifier for the JWT. Id attribute verifies if the JWT contains the specific jti claim. When the text value and ref attribute are both empty, the policy generates a jti containing a random UUID.</p> <pre data-bbox="659 1361 1396 1518"> &lt;Id&gt;explicit-jti&lt;/Id&gt; -or- &lt;Id ref='varname' /&gt; -or- &lt;Id/&gt; </pre>
IgnoreUnresolvedVariables (Optional)	<p data-bbox="659 1552 1358 1641">Set to false if you want the policy to throw an error when any referenced variable specified in the policy is unresolvable. Set to true to treat any unresolvable variable as an empty string</p> <pre data-bbox="659 1664 1396 1753"> &lt;IgnoreUnresolvedVariables&gt;true false&lt;/IgnoreUnresolvedVariables&gt; </pre>
Issuer (Optional)	<p data-bbox="659 1787 1396 1843">The policy verifies that the issuer in the JWT matches the string specified in the configuration element.</p> <pre data-bbox="659 1865 1396 1957"> &lt;Issuer ref='variable-name-here' /&gt; &lt;Issuer&gt;issuer-string-here&lt;/Issuer&gt; </pre>

## Elements and Attributes

## Description

<PublicKey/JWKS> (JWKS or Value element is required to verify a JWT signed with RSA algorithm)

Specifies a value in the JSON web key set (JWKS) format containing a set of public keys. If the JWT contains a key ID in the set of JWKS, then the policy uses the correct public key to verify the JWT.

```
<PublicKey>
 <JWKS ref="public.jwks" />
</PublicKey>
or
<PublicKey>
 <JWKS>JWKS-value</JWKS>
</PublicKey>
```

<PublicKey/Value> (JWKS or Value element is required to verify a JWT signed with RSA algorithm)

Use this element only when the algorithm is an RSA variant. This element specifies the public key to verify the signature. Specify the PEM-encoded key directly or use the ref attribute to pass the key in a flow variable.

```
<PublicKey>
 <Value ref="public.publickey" />
</PublicKey>
-or-
<PublicKey>
 <Value>
 -----BEGIN PUBLIC KEY-----

MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAw2kP
rRzcufvUNHvTH/WW

Q0UrCw5c0+Y707KX3PpXkZGbtTT4nvU1jC0d1lHV8MfUyRXm
pmnNxJHAC2F73IyN

C5TBtXMORc+us7A2cTtC4gZV256bT4h3sIEMsDl0Joz9K9MP
zVPFxa1i0RgNt06n
 Xn/
Bs2UbbLlKP5Q1HPxewUDEh0gVMqz9wdIGwH1pPxKvd3NltYG
fPsUQovlof3l2

ALvO7i5Yrm96kknfFEWf1EjmCCKvz2vjVbBb6mp1ZpYfc9MO
TZVpQcXSbzb/BWUo
 ZmkDb/
DRW5onc1GzxQITBFP3S6JXd4LNEsJcTp705ec1cQ9Wp2Kl+n
KrKyv1E5Xx
 DQIDAQAB
 -----END PUBLIC KEY-----
 </Value>
</PublicKey>
```

### ⚠ Restriction

JWT validation fails RSA keys smaller than 2048 bits. Ensure that your keys are 2048 bits or larger.

<Source> (Optional)

If present, specifies the flow variable in which the policy expects to find the JWT to verify.

```
<Source>jwt-variable</Source>
```

Elements and Attributes	Description
<Subject> (Optional)	The policy verifies that the subject in the JWT matches the string specified in the policy configuration.  <pre>&lt;Subject&gt;subject-string-here&lt;/Subject&gt;</pre>
<SecretKey/Value> (Optional)	Provides the secret key used to verify or sign tokens with an HMAC algorithm. Use only when the algorithm is one of HS256, HS384, HS512. Use the ref attribute to pass the key in a flow variable.  <pre>&lt;SecretKey&gt;   &lt;Value ref="private.your-variable-name" /&gt; &lt;/SecretKey&gt;</pre>
<TimeAllowance> (Optional)	The "grace period" for times. For example, if the time allowance is configured to be 60s, then an expired JWT would be treated as still valid, for 60s after the asserted expiry. The not-before-time will be evaluated similarly. Default value is 0s.  <pre>&lt;TimeAllowance&gt;60s&lt;/TimeAllowance&gt;</pre>

The following flow variables are available after the policy is executed:

#### Flow Variables

Variable	Description
header.algorithm	Signing algorithm used on JWT.
claim.subject	JWT subject claim.
claim.issuer	JWT issuer claim.
claim.audience	JWT audience claim. This value may be a string, or an array of strings.
claim.expiry	Expiration date or time, expressed in seconds.
expiry_formatted	Expiration date or time, formatted as a human readable string. Example: 2019-18-28T21:30:45.000+0000
seconds_remaining	Number of seconds before the token expires. If the token is expired, this number will be negative.
time_remaining_formatted	Time remaining before the token expires, formatted as a human-readable string. Example: 00:59:59.926
is_expired	true or false
claim.issuedat	Token issued Date, expressed in seconds since epoch.
claim.notbefore	If the JWT includes a nbf claim, this variable will contain the value. This is expressed in seconds since epoch.
valid	In the case of VerifyJWT, this variable will be true when the signature is verified, and the current time is before the token expiry, and after the token notBefore value, if they are present. Otherwise false.  In the case of DecodeJWT, this variable is not set.

Variable	Description
claim.name	The value of the named claim (standard or additional) in the payload. One of these will be set for every claim in the payload.
header.name	The value of the named header (standard or additional). One of these will be set for every additional header in the header portion of the JWT.
header.kid	The Key ID, if added when the JWT was generated.
header.type	Will be set to JWT.
payload-claim-names	An array of claims supported by the JWT.
payload-json	Payload in JSON format.
header-json	Header in JSON format.

During the policy execution, the following errors can occur:

Error Cause

Error Name	Cause
steps.jwt.AlgorithmInTokenNotPresentInConfiguration	Occurs when the verification policy has multiple algorithms.
steps.jwt.AlgorithmMismatch	The algorithm specified in the Generate policy did not match the one expected in the Verify policy. The algorithms specified must match
steps.jwt.FailedToDecode	The policy was unable to decode the JWT. The JWT is possibly corrupted.
steps.jwt.GenerationFailed	The policy was unable to generate the JWT.
steps.jwt.InsufficientKeyLength	For a key less than 32 bytes for the HS256 algorithm, less than 48 bytes for the HS386 algorithm, and less than 64 bytes for the HS512 algorithm.
steps.jwt.InvalidClaim	For a missing claim or claim mismatch, or a missing header or header mismatch.
steps.jwt.InvalidCurve	The curve specified by the key is not valid for the Elliptic Curve algorithm.
steps.jwt.InvalidJsonFormat	Invalid JSON found in the header or payload.
steps.jwt.InvalidToken	This error occurs when the JWT signature verification fails.
steps.jwt.JwtAudienceMismatch	The audience claim failed on token verification.
steps.jwt.JwtIssuerMismatch	The issuer claim failed on token verification.
steps.jwt.JwtSubjectMismatch	The subject claim failed on token verification.
steps.jwt.KeyIdMissing	The Verify policy uses a JWKS as a source for public keys, but the signed JWT does not include a kid property in the header
steps.jwt.KeyParsingFailed	The public key could not be parsed from the given key information.
steps.jwt.NoAlgorithmFoundInHeader	Occurs when the JWT contains no algorithm header.
steps.jwt.NoMatchingPublicKey	The Verify policy uses a JWKS as a source for public keys, but the kid in the signed JWT is not listed in the JWKS.

Error Name	Cause
steps.jwt.SigningFailed	In GenerateJWT, for a key less than the minimum size for the HS384 or HS512 algorithms
steps.jwt.TokenExpired	The policy attempts to verify an expired token.
steps.jwt.TokenNotYetValid	The token is not yet valid.
steps.jwt.UnknownException	An unknown exception occurred.
steps.jwt.WrongKeyType	Wrong type of key specified. For example, if you specify an RSA key for an Elliptic Curve algorithm, or a curve key for an RSA algorithm.

The following fault variables are set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
fault.name="fault_name"	fault_name is the name of the fault, as listed in the Runtime errors table above. The fault name is the last part of the fault code.	fault.name Matches "TokenExpired"
jwt.policy_name.failed	policy_name is the user-specified name of the policy that threw the fault.	jwt.JWT-Policy.failed = true

Following is an example of a fault rule:

#### Sample Code

```
<FaultRules>
 <FaultRule name="JWT Policy Errors">
 <Step>
 <Name>JavaScript-1</Name>
 <Condition>(fault.name Matches "TokenExpired")</Condition>
 </Step>
 <Condition>jwt.JWT-1.failed=true</Condition>
 </FaultRule>
</FaultRules>
```

## Related Information

[JSON Web Tokens \[page 300\]](#)

[Generate JWT \[page 302\]](#)

[Decode JWT \[page 319\]](#)

### 1.5.1.4.1.13.3 Decode JWT

This policy describes about Decode JSON Web Token(JWT) Policy.

This policy verifies a signed JWT, with a configurable set of claims. When this policy executes, API Management verifies the signature of a JWT, and verifies that the JWT is valid according to the expiry and not-before times if they are present. The policy can optionally also verify the values of specific claims on the JWT, such as the subject, the issuer, the audience, or the value of additional claims. If the JWT is verified and valid, then all of the claims contained within the JWT are extracted into context variables for use by subsequent policies or conditions, and the request is allowed to proceed. If the JWT signature cannot be verified or if the JWT is invalid because of one of the timestamps, all processing stops and an error is returned in the response.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

The policy shown below decodes a JWT found in the flow variable var.jwt. This variable must be present and contain a viable (decodable) JWT. The policy can obtain the JWT from any flow variable.

```

<Code Syntax>
<DecodeJWT async=\"false\" continueOnError=\"false\" enabled=\"true\"
xmlns=\"http://www.sap.com/apimgmt\">
 <Source>var.jwt</Source>
</DecodeJWT>

```

Following table lists the elements and attributes that you can configure on this policy

Elements and Attributes	Description
<Source> (Optional)	If present, specifies the flow variable in which the policy expects to find the JWT to decode.
	<code>&lt;Source&gt;jwt-variable&lt;/Source&gt;</code>

The following flow variables are available after the policy is executed:

Flow Variables

Variable	Description
header.algorithm	Signing algorithm used on JWT.
claim.subject	JWT subject claim.
claim.issuer	JWT issuer claim.
claim.audience	JWT audience claim. This value may be a string, or an array of strings.

Variable	Description
claim.expiry	Expiration date or time, expressed in seconds.
expiry_formatted	Expiration date or time, formatted as a human readable string. Example: 2019-18-28T21:30:45.000+0000
seconds_remaining	Number of seconds before the token expires. If the token is expired, this number will be negative.
time_remaining_formatted	Time remaining before the token expires, formatted as a human-readable string. Example: 00:59:59.926
is_expired	true or false
claim.issuedat	Token issued Date, expressed in seconds since epoch.
claim.notbefore	If the JWT includes a nbf claim, this variable will contain the value. This is expressed in seconds since epoch.
valid	In the case of VerifyJWT, this variable will be true when the signature is verified, and the current time is before the token expiry, and after the token notBefore value, if they are present. Otherwise false.  In the case of DecodeJWT, this variable is not set.
claim.name	The value of the named claim (standard or additional) in the payload. One of these will be set for every claim in the payload.
header.name	The value of the named header (standard or additional). One of these will be set for every additional header in the header portion of the JWT.
header.kid	The Key ID, if added when the JWT was generated.
header.type	Will be set to JWT.
payload-claim-names	An array of claims supported by the JWT.
payload-json	Payload in JSON format.
header-json	Header in JSON format.

During the policy execution, the following errors can occur:

Error Cause

Error Name	Cause
steps.jwt.FailedToDecode	Occurs when the policy is unable to decode the JWT. The JWT may be malformed, invalid or otherwise not decodable.
steps.jwt.InvalidToken	Occurs when the flow variable specified in the <Source> element of the policy is out of scope or can't be resolved.

The following fault variables are set when the policy triggers an error at runtime:



## Fault Variables

Variable Set	Where	Example
fault.name="fault_name"	fault_name is the name of the fault, as listed in the Runtime errors table above. The fault name is the last part of the fault code.	fault.name Matches "TokenExpired"
jwt.policy_name.failed	policy_name is the user-specified name of the policy that threw the fault.	jwt.JWT-Policy.failed = true

Following is an example of a fault rule:

### Sample Code

```
<FaultRules>
 <FaultRule name="JWT Policy Errors">
 <Step>
 <Name>JavaScript-1</Name>
 <Condition>(fault.name Matches "TokenExpired")</Condition>
 </Step>
 <Condition>jwt.JWT-1.failed=true</Condition>
 </FaultRule>
</FaultRules>
```

## Related Information

[JSON Web Tokens \[page 300\]](#)

[Generate JWT \[page 302\]](#)

[Verify JWT \[page 311\]](#)

### 1.5.1.4.1.14 Key Value Map Operations

The Key Value Map Operations policy allows you to create a key value map and perform update, read, and delete operations on the map.

Key Value Map Operations are typically used to store or retrieve long-lived information that needs to be reused over multiple request or response transactions.

KeyValue refers to any arbitrary data in the format key=value, for example localhost=127.0.0.1,zip\_code=94110, or first\_name=Philip.

In the first example, localhost is a key, and 127.0.0.1 is a value.

Each key/value pair is stored in a map as an entry. A key/value map can store many entries. For example, say you need to store a list of IP addresses associated with various backend environments.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

### Code Syntax

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<KeyValueMapOperations async="true" continueOnError="false"
 enabled="true" mapIdentifier="urlMapper" xmlns="http://www.sap.com/
 apimgmt">
 <InitialEntries>
 <Entry>
 <Key>
 <Parameter>key1</Parameter>
 </Key>
 <Value>val1</Value>
 </Entry>
 <Entry>
 <Key>
 <Parameter>var_name</Parameter>
 </Key>
 <Value>val1</Value>
 <Value>val2</Value>
 </Entry>
 </InitialEntries>
 <Put override="false">
 <Key>
 <Parameter>key1</Parameter>
 </Key>
 <Value ref="var_name"/>
 </Put>
 <Get assignTo="sapapim.empnumber" index="1">
 <Key>
 <Parameter ref="sapapim.empEmail"/>
 </Key>
 </Get>
 <Delete>
 <Key>
 <Parameter>key1</Parameter>
 </Key>
 </Delete>
 <Scope>apiproxy</Scope>
</KeyValueMapOperations>
```

### <mapIdentifier> element

Element	Default	Type	Description
mapIdentifier (Optional)	N/A	N/A	Specifies an identifier to be used when accessing a map created by this policy.

Element	Default	Type	Description
			If you exclude this attribute, a KVM named <code>kvmmap</code> is used.

### <Scope> element

Element	Default	Type	Description
Scope (Optional)	environment	String	<p>Defines the boundary of accessibility for Key Value Maps.</p> <p>The default value is environment. That is, by default, maps entries are shared by all API proxies running in an environment.</p> <p>The valid values are <code>apiproxy</code>, <code>environment</code>, <code>organization</code> and <code>policy</code>.</p> <p>If you set the scope to <code>apiproxy</code>, then the entries in the key value map are accessible only by the API Proxy that writes the values to the map.</p>

#### Note

when accessing a map or map entry, you must specify the same scope value you used when the map was created. For example, if the map was created with a scope of `apiproxy`, you must use the `apiproxy` scope when retrieving its values, updating changes, or deleting entries.

### <Entry> element

Element	Default	Type	Description
Entry (Optional)	N/A	N/A	Seed values for key value maps, which are populated in the key value map when it's initialized.

Element	Default	Type	Description
			<p>↔ Sample Code</p> <pre> &lt;InitialEntries&gt;   &lt;Entry&gt;     &lt;Key&gt;      &lt;Parameter&gt;key1&lt;/Parameter&gt;     &lt;/Key&gt;      &lt;Value&gt;val1&lt;/Value&gt;   &lt;/Entry&gt;   &lt;Entry&gt;     &lt;Key&gt;      &lt;Parameter&gt;key1_variable&lt;/Parameter&gt;     &lt;/Key&gt;      &lt;Value&gt;val2&lt;/Value&gt;      &lt;Value&gt;val3&lt;/Value&gt;   &lt;/Entry&gt; &lt;/InitialEntries&gt; </pre>

### <InitialEntries> element

Element	Default	Type	Description
InitialEntries (Optional)	N/A	N/A	<p>Seed values for key value maps, which are populated in the key value map when it's initialized. Make sure to specify the name of the KVM with the mapIdentifier attribute on the parent element.</p> <p>When using this element, when you save the policy on a deployed version of the proxy, or deploy the API proxy bundle containing the policy with this element, the key(s) are automatically created in the KVM (as unencrypted). If the values</p>

Element	Default	Type	Description
---------	---------	------	-------------

in the policy are different than the values in the KVM, the values in the KVM are overwritten when the proxy is deployed. Any new keys/values are added to the existing KVM alongside the existing keys/values.

#### Sample Code

```
<InitialEntries>
 <Entry>
 <Key>

 <Parameter>key1</Parameter>
 </Key>

 <Value>val1</Value>
 </Entry>
 <Entry>
 <Key>

 <Parameter>key2_variable</Parameter>
 </Key>

 <Value>val2</Value>

 <Value>val3</Value>
 </Entry>
</InitialEntries>
```

#### Note

Keys and values populated by this element must be literals.

#### Sample Code

```
<Parameter ref="request.queryparam.key">
```

Element	Default	Type	Description
			The example shown above isn't supported within this element.

### <key> element

Element	Default	Type	Description
key (Optional)	N/A	N/A	Specifies the key in a key/value map entry. A key can be composite, which means that more than one parameter can be appended to create the key. For example, userID and role might be combined to create a key.

↔ Sample Code

```
<Key>
<Parameter>key
1</Parameter>
</Key>
```

See the <parameter> element for information about how to set the key name.

### <parameter> element

Element	Default	Type	Description
parameter (Required)	N/A	String	Specifies the key in a key/value pair. This element specifies the name when creating, putting, retrieving, or deleting the key/value pair. <ul style="list-style-type: none"> <li>A literal string</li> </ul>

↔ Sample Code

```
<Key>
```

Element	Default	Type	Description
---------	---------	------	-------------

```
<Parameter>
stringliter
al</
Parameter>
</Key>
```

- A variable to be retrieved at runtime

#### ↔ Sample Code

```
<Key>

<Parameter
ref="var_na
me" />
</Key>
```

- A combination of variable references and literal strings

#### ↔ Sample Code

```
<Parameter>
targetendpo
int</
Parameter>

<Parameter
ref="api_pr
oxy.name" />

<Parameter>
size</
Parameter>
</Key>
```

When the Key element includes multiple Parameter elements, the effective key string is the concatenation of the values of each parameter, joined with a double underscore. For example, in the above example, if the api\_proxy.name variable has the value "def23", then the

Element	Default	Type	Description
			effective key will be targetendpoint_def23_size .

**Note**

Whether you're retrieving, updating, or deleting a key/value entry, the key name must match the name of the key in the key value map.

#### ref attribute of the <parameter> element

Attribute	Default	Type	Description
ref	N/A	N/A	Specifies the name of a variable whose value contains the exact name of the key you want to create, retrieve, or delete.

#### <value> element

Element	Default	Type	Description
value (Required)	N/A	String	Specifies the value of a key.  You can specify the value as either a literal string or, using the ref attribute, as a variable to be retrieved at runtime.

#### Sample Code

```
<!-- Specify a literal value -->
<Value>literal string<Value>
```

#### Sample Code

```
<!-- Specify the name of variable value to be populated at run time. -->
```



Element	Default	Type	Description
---------	---------	------	-------------

```
<Value
 ref="var_name"
 />
```

You can include multiple <value> elements to specify a multi-part value. Values are combined at run time.

In the following example, two keys 'key1' with values 'val1' and 'val2' and 'key2' with values 'val3' and 'val4' are added to the KVM:

#### Sample Code

```
<InitialEntries>
 <Entry>
 <Key>
 <Parameter>key1</Parameter>
 </Key>
 <Value>val1</Value>
 <Value>val2</Value>
 </Entry>
 <Entry>
 <Key>
 <Parameter>key2</Parameter>
 </Key>
 <Value>val3</Value>
 <Value>val4</Value>
 </Entry>
</InitialEntries>
```

#### ref attribute of the <value> element

Attribute	Default	Type	Description
ref	N/A	N/A	Specifies the name of a variable whose value

Attribute	Default	Type	Description
			contains the key value(s) you want to set.

### <Get> element

Element	Description	Attributes	Description
Get (Required if <Put> or <Delete> are not present)	Retrieves the value for the key specified.  At least one of <Get>, <Put>, or <Delete> must be used.  Make sure to specify the name of the KVM with the mapIdentifier attribute on the parent element.	assignTo  index	The variable to which the retrieved value must be assigned.  You can specify an index for a multivalued key. For example, if index=1, the value at index 1 is fetched and assigned to the variable. If not specified, all the values of that entry are assigned to the variable as java.util.List.
	<pre> ↳ Sample Code  &lt;Get   assignTo="var5   "   index="1"&gt;    &lt;Key&gt;    &lt;Parameter&gt;key   1&lt;/   Parameter&gt;    &lt;/   Key&gt; &lt;/Get&gt; </pre>		

### <Put> element

Element	Description	Attributes	Description
Put (Required if <Get> or <Delete> are not present)	Writes a key/value pair to a key value map.	override	The default value is false.  If true, it overrides the value of a key.
	<pre> ↳ Sample Code  &lt;Put   override="false"&gt;    &lt;Key&gt;    &lt;Parameter&gt; </pre>		

Element	Description	Attributes	Description
	<pre>ref="mykeyvariable" /&gt; &lt;/ Key&gt; &lt;Value ref="myvalvariable1" /&gt; &lt;/Put&gt;</pre>		

### <Delete> element

Element	Default	Type	Description
Delete (Required if <Get> or <Put> are not present)	N/A	N/A	<p>Deletes the specified key/value pair. At least one of &lt;Get&gt;, &lt;Put&gt;, or &lt;Delete&gt; must be used.</p> <p>Make sure to specify the name of the KVM with the 'mapIdentifier' attribute on the parent element.</p> <div data-bbox="1107 1025 1396 1317" data-label="Code-Block"> <pre>Sample Code  &lt;Delete&gt;   &lt;Key&gt;    &lt;Parameter&gt;key1&lt;/Parameter&gt;   &lt;/Key&gt; &lt;/Delete&gt;</pre> </div>

During the policy execution, the following errors can occur:

## Error Cause

Error Name	Cause
SetVariableFailed	This error occurs when you attempt to fetch a value from an encrypted key-value map and assign it to a variable that doesn't have the "private" prefix in its name. This prefix is necessary for basic security measures during debugging, as it prevents the encrypted values from being displayed in API proxy Trace and debug sessions. For example: <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>↔ Sample Code</p> <pre>&lt;Get   assignTo="private.encryptedVar"   index="1"&gt;   &lt;Key&gt;     &lt;Parameter&gt;foo&lt;/Parameter&gt;   &lt;/Key&gt; &lt;/Get&gt;</pre> </div>
RemoveVariableFailed	Failed to remove variable {0} in KeyValueCollectionDefinition {1}
InvalidIndex	Invalid index {0} in KeyValueCollectionDefinition {1}
KeysMissing	Key element is missing in KeyValueCollectionDefinition {0}
ValuesMissing	Value element is missing in KeyValueCollectionDefinition {0}

## 1.5.1.4.1.15 Message Logging Policy

The Message Logging policy lets you send syslog messages to third-party log management services, such as Splunk, Sumo Logic, Loggly, or similar log management services.

If you want to send syslog to one of those services, follow the service documentation of the concerned service to obtain the host, port, and protocol, then set the <Syslog> element on this policy accordingly.

You can attach this policy in the following locations:

ProxyEndpoint					TargetEndpoint			← Response
Request →	PostClientFlow	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
		•	•	•	•	•	•	
	•	•	•	•	•	•	•	
	PostClientFlow	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

```
<MessageLogging async="false" continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
 <Syslog>
 <Message>Message.id = {request.header.id}</Message>
 <!-- Host must be valid DNS/IP -->
 <!-- <Host>127.0.0.1</Host> -->
 <Host></Host>
```

```

<!-- This is default port value -->
<Port>514</Port>
<Protocol>TCP</Protocol>
<SSLInfo>
 <Enabled>>false</Enabled>
 <ClientAuthEnabled>>false</ClientAuthEnabled>
 <KeyStore/>
 <KeyAlias/>
 <TrustStore/>
 <Ciphers/>
 <Protocols/>
</SSLInfo>
</Syslog>
</MessageLogging>

```

### Note

When using Loggly, `<FormatMessage>>true</FormatMessage>` is required in the policy as a child of the `<Syslog>` element.

Elements and Attributes	Description
<p><b>Syslog destination</b></p> <p>To send syslog to Splunk, Sumo Logic, Loggly, or similar log management services.</p>	<p><b>Message</b></p> <p>Build the message to be sent to the syslog, combining text with variables to capture the information you want.</p> <div data-bbox="1021 1041 1394 1332" data-label="Text"> <p><b>Note</b></p> <p>Response variables will not be available in PostClientFlow following an Error Flow. Use message variables to log response information for both error and success situations.</p> </div>
<b>Host</b>	The hostname or IP address of the server where the syslog should be sent. If the element is not specified, the default is localhost.
<b>Port</b>	Port where the syslog is running. If you don't include this element, the default is 514.
<b>Protocol</b>	TCP or UDP (default). While UDP is more performant, the TCP protocol guarantees message log delivery to the syslog server. For sending syslog messages over TLS/SSL, only TCP is supported.

## Elements and Attributes

## Description

FormatMessage

true or false

Optional, but <FormatMessage>true</FormatMessage> is required for use with Loggly.

This element lets you control the format of generated content prepended to the message. If set to true, the syslog message is prepended by a fixed number of characters, which lets you filter out that information from messages.

## Elements and Attributes

## Description

SSLInfo

If you don't include this element or leave it empty, the default value is false.

Lets you log messages through SSL/TLS. Use with sub elements

- Enabled: Indicates whether TLS/SSL is enabled for the endpoint. The default value is false.
- ClientAuthEnabled: Outbound client authentication (2-way TLS/SSL)
- Keystore: A keystore containing private keys used for outbound client authentication
- KeyAlias: The key alias of the private key used for outbound client authentication
- TrustStore: A keystore containing trusted server certificates.
- Ciphers: Supported ciphers for outbound TLS/SSL.To restrict ciphers, add the following elements listing the supported ciphers:

### Sample Code

```
<Ciphers>
 <Cipher>TLS_RSA_WITH_3DES_EDE_CBC_SHA
 </Cipher>
 <Cipher>TLS_RSA_WITH_DES_CBC_SHA</Cipher>
</Ciphers>
```

- Protocols: Supported protocols for outbound TLS/SSL.To restrict protocols, add the following elements listing the supported protocols:

### Sample Code

```
<Protocols>
 <Protocol>TLSv1</Protocol>
```

```
<Protocol>TLSv1.2<
/Protocol>

<Protocol>SSLv2Hello</Protocol>
</Protocols>
```

#### Sample Code

```
<SSLInfo>

<Enabled>>false</
Enabled>

<ClientAuthEnabled>fa
lse</
ClientAuthEnabled>

<KeyStore>myKeystore<
/KeyStore>

<KeyAlias>myKey</
KeyAlias>

<TrustStore>myTrustst
ore</TrustStore>
</SSLInfo>
```

## 1.5.1.4.1.16 Quota

The Quota policy defines the number of request messages an application can submit to an API over a given period of time.

The period of time can be an hour, a day, or a month and so on. You can apply this policy on the context of request messages.

The Quota policy helps API Providers to restrict the number of calls made to an API. For example, you can restrict access to your applications by defining the number of API calls made as 20 per day or 20,000 over a period of one week.

API Management maintains a counter that keeps track of the number of calls made to the API. Once the counter reaches the Quota limit, all successive calls made to the API are rejected. API Management returns an error message to an API-call made after the Quota limit is exceeded. The Quota policy provides the capability to reset the counters automatically after the stipulated period of time, unless it is explicitly reset by using the Reset Quota policy.



## Related Information

[Types of Quota \[page 337\]](#)

[Static and Dynamic Settings \[page 338\]](#)

[Designing Quota Policy \[page 339\]](#)

### 1.5.1.4.1.16.1 Types of Quota

Quota type is an attribute of the Quota policy that you define while configuring the policy. API Management supports the following three types of Quota:

- `calendar` type of Quota. For example, `<Quota name="DemoQuota" type="calendar">`  
In the calendar Quota, you explicitly provide a start time. The counter starts the count from the specified start date. The Quota counter is refreshed based on the Interval and TimeUnit that you set. For example, the following Quota of type calendar begins counting at 8.30 am on June 26, 2015, and will refresh once in every 20 minutes.

#### Sample Code

```
<Quota type="calendar">
 <Identifier ref="request.header.clientId"/>
 <StartTime>2015-06-26 08:30:00</StartTime>
 <Interval>20</Interval>
 <TimeUnit>minute</TimeUnit>
 <Allow count="99"/>
 <MessageWeight ref="request.header.weight"/>
 <Distributed>true</Distributed>
 <Synchronous>true</Synchronous>
</Quota>
```

#### Note

If the type is not mentioned, then the Quota defaults to `calendar` type.

**rollingwindow:** A "rolling window" counter advances by the time interval that you specify. You do not specify a `StartTime`; instead, the `StartTime` for the counter is the time when the first message is received from the client plus the interval that you define. A counter is kept for each client ID (consumer key). Thus, the counter will reset to zero when the Interval you define has passed. This enables you to configure a Quota in which an app is indefinitely allowed 1000 requests every 24 hours.

**flexi:** Flexible Quota enforcement causes the counter to begin when the first request message is received from an app. Under flexible Quota enforcement, `StartTime` is dynamic; every app has its own `StartTime` based on the time when the first request is received. This enables you to provide Quotas that support one week, one month, or 6 months access to your API, customized for each app.

## Related Information

[Static and Dynamic Settings \[page 338\]](#)

## 1.5.1.4.1.16.2 Static and Dynamic Settings

A Quota can be static or dynamic.

### Static Settings

For a static quota, you provide a count, a time interval, and a time unit. In the example below, the application accepts 5000 requests per week.

#### Sample Code

```
<Quota>
 <Allow count="5000" />
 <Interval>1</Interval>
 <TimeUnit>week</TimeUnit>
</Quota>
```

### Dynamic Settings

Dynamic Quotas enable you to configure a single Quota policy that enforces different Quota settings for different applications; based on the identity of the requesting application. Dynamic Quota values are populated at runtime by resolving an application identifier to an API product. The Identifier can be a field in the request that is unique to each application. For API proxies where OAuth is enforced, you can use `consumer_id` as the Identifier, as demonstrated in the sample policy below:

#### Sample Code

```
<Quota>
 <Intervalref="apiproduct.developer.quota.interval"/>
 <TimeUnit ref="apiproduct.developer.quota.timeunit"/>
 <Allow countRef="apiproduct.developer.quota.limit"/>
 <Identifier ref=" consumer_id"/>
</Quota>
```

The example above uses the variable `consumer_id` to identify the requesting application. This works as long as the request message contains a `consumer_id` (associated with an OAuth-enabled request).

### Related Information

## 1.5.1.4.1.16.3 Designing Quota Policy

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	←Response
	•						
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

### Code Syntax

```
<Quota type="calendar" async="true" continueOnError="true" enabled="true">
 <Identifier ref="{ref}"></Identifier>
 <MessageWeight ref="{ref}"></MessageWeight>
 <Allow count="{count}" countRef="{countref}">
 </Allow>
 <SyncIntervalInSeconds>{value}</SyncIntervalInSeconds>
 <SyncMessageCount>{count}</SyncMessageCount>
 </AsynchronousConfiguration>
 <Interval ref="{ref}" type="{type}"></Interval>
 <Distributed>true</Distributed>
 <PreciseAtSecondsLevel>true</PreciseAtSecondsLevel>
 <StartTime>{time}</StartTime>
 <Synchronous>true</Synchronous>
 <TimeUnit ref="{ref}" type="{type}"></TimeUnit>
</Quota>
```

Elements & Attributes	Description
StartTime (Mandatory)	Use this attribute only for Quota policies of type calendar. Indicates the date and time when the Quota counter begins counting (regardless of whether or not any requests have been received from any applications.) This value is in UTC.  Valid value: date and time, for example 2015-02-09 00:00:00.
Interval (Mandatory)	Specifies the interval of time (in hours, minutes, or days as defined by TimeUnit) applicable to the Quota. For example, a value of 24 for the Interval with a TimeUnit of hours means that the Quota is calculated over one day (24 hours).  Valid value: integer  The ref attribute identifies the variable that provides the value of the Interval.
TimeUnit (Mandatory)	Valid values: second, minute, hour, day, or month  The ref attribute identifies the variable that provides the value of the TimeUnit.
Allow (Mandatory)	Specifies the maximum number of inbound requests.

Elements & Attributes	Description
count (Optional)	<p>Specifies a message count for the quota.</p> <p>For example, a value of 200 for the Allow count with duration of 1 month means that the quota is set to be 200 messages per month.</p> <p>Valid value: integer</p> <p>Default Value: 2000</p>
countRef (Optional)	<p>This attribute identifies the variable that provides the value of the Quota limit.</p> <p>If a count reference is specified, it takes precedence over the Allow count value.</p> <p>Example:</p> <p>The element <code>Allow count="2000"</code></p> <p><code>countRef="request.header.allowed_quota"/&gt;</code> has a <code>count header (countRef="request.header.allowed_quota" )</code> along with the count value of 2000.</p>
Identifier (Optional)	<p>The variable used to uniquely identify the client application is referred to as the "Identifier". The "ref" attribute is used to specify the variable that contains the value of the Identifier. If the "ref" attribute is not used, the policy will utilize a single counter that is applied to the quota.</p>
MessageWeight (Optional)	<p>Specifies the weight defined for each message.</p> <p>Message weight is used to increase impact of request messages that, for example, consume more computational resources than others. For example, you may want to calculate POST messages as being twice as expensive as GET messages. A value representing MessageWeight can be extracted from HTTP headers, query parameters, or an XML or JSON request payload.</p>
Distributed	<p>When set to true, a central counter is maintained that is continuously updated by all API Management servers.</p> <p>Note: Always set this value to true.</p>
PreciseAtSecondsLevel	<p>The default precision for Quotas intervals is one minute. When set to true, Quota precision is set to record at intervals of one second. Use this setting when you have a Quota that uses minutes as the TimeUnit, and you need to ensure that Quotas are counted and enforced by seconds.</p> <p>Valid values: true or false</p>
Synchronous	<p>This setting determines how the distributed Quota counter is updated. If set to true, the quota counter is updated in the central repository synchronously. This means that the update is made at the same time the API call is quota-checked. When synchronous is set to true, you are guaranteed that no API calls over the quota will be allowed.</p> <p>Note: Always set this value to true.</p>

Elements & Attributes	Description
AsynchronousConfiguration(Optional)	<p>This configuration element is only required when Synchronous is set to false. This element enables you to configure the synchronization interval among distributed Quota counters. The default synchronous update interval is 10 seconds. You can modify this by adding the child element SyncIntervalInSeconds.</p> <p>Example</p> <div data-bbox="603 555 1394 770" style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p><b>Sample Code</b></p> <pre data-bbox="644 636 1356 734">&lt;Synchronous&gt;false&lt;/Synchronous&gt; &lt;AsynchronousConfiguration&gt;   &lt;SyncIntervalInSeconds&gt;15&lt;/SyncIntervalInSeconds&gt; &lt;/AsynchronousConfiguration&gt;</pre> </div> <p>Alternatively, you can use the SyncMessageCount option instead, but you cannot use both. SyncMessageCount specifies the number of requests across all API Management message processors between quota updates. The following example specifies that the quota count is updated every 10 requests across all API Management message processors:</p> <div data-bbox="603 976 1394 1191" style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p><b>Sample Code</b></p> <pre data-bbox="644 1057 1209 1155">&lt;Synchronous&gt;false&lt;/Synchronous&gt; &lt;AsynchronousConfiguration&gt;   &lt;SyncMessageCount&gt;10&lt;/SyncMessageCount&gt; &lt;/AsynchronousConfiguration&gt;</pre> </div>

## Related Information

- [Quota \[page 336\]](#)
- [Reset Quota \[page 343\]](#)
- [Spike Arrest \[page 353\]](#)

### 1.5.1.4.1.17 Raise Fault

The RaiseFault policy allows you to create custom messages in case of error conditions. This policy returns a FaultResponse to the requesting application if it encounters an error condition.

A FaultResponse can consist of HTTP headers, query parameters, and a message payload. These elements can be populated using variables. This enables you to send customized FaultResponses that are specific to the error conditions.

During execution, the RaiseFault policy transfers the message flow to the default ErrorFlow, which in turn returns the designated FaultResponse to the requesting application. When the message flow switches to the

default ErrorFlow, no further policy processing occurs. All remaining processing steps are bypassed, and the FaultResponse is returned directly to the requesting app.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows::

```

Code Syntax

<!-- The policy will create a custom response of status code as 500 and
message as Server error in case of error condition is met -->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RaiseFault async="true" continueOnError="false" enabled="true" xmlns="http://
www.sap.com/apimgmt">
 <FaultResponse>
 <Set>
 <Headers/>
 <Payload contentType="text/plain"> </Payload>
 <StatusCode>500</StatusCode>
 <ReasonPhrase>Server Error</ReasonPhrase>
 </Set>
 </FaultResponse>
 <IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
</RaiseFault>

```

Field Name	Description
IgnoreUnresolvedVariables(Optional)	Ignores any unresolved variable error in the flow.  Valid values: true or false  Default value: true
FaultResponse(Optional)	Defines the response message returned to the requesting application.

The following predefined flow variables are available after Raise Fault policy executes.

Flow Variables

Variable	Type	Permission	Description
fault.name	String	Read-Only	Returns the fault name in the error and if not available, an empty string.
fault.type	String	Read-Only	Returns the fault type in the error and if not available, an empty string.

Variable	Type	Permission	Description
fault.category	String	Read-Only	Returns the fault category in the error and if not available, an empty string.

During the policy execution, the following error can occur:

Error Cause

Error Name	Cause
RaiseFault	See fault string.

Following fault variables are set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	The [prefix] is raisefault.  The [policy_name] is the name of the policy that threw the error.	raisefault.RF-ThrowError.failed = true
fault.[error_name]	[error_name] = The specific error name to check for as listed in the table above.	fault.name = "RaiseFault"

## 1.5.1.4.1.18 Reset Quota

The Reset Quota policy enables you to reset the limit for a specified Quota policy.

For example, you can use this policy to reset a Quota counter when additional API calls are to be made. Attach this policy before the Quota policy that you intend to reset.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	←Response
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

### ↔ Code Syntax

```
<!-- The policy will reset the Quota policy by 100 calls when triggered, if
the quota for a user is over when this policy is triggered this will allow
user to make another 100 calls -->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ResetQuota async="true" continueOnError="false" enabled="true" xmlns="http://
www.sap.com/apimgmt">
```

```

<Quota name="impose-quota">
 <Identifier ref="request.queryparam.apiKey">
 <Allow>100</Allow>
 </Identifier>
</Quota>
</ResetQuota>

```

Elements and Attributes	Description
Quota (Mandatory)	<p>Specifies the name of the Quota policy whose counter should be reset.</p> <p>The ref attribute identifies the variable that fetches the Quota policy that has to be reset.</p> <p>Both name and ref are optional but at least one of attributes should be used.</p>
Identifier (Optional)	<p>Specifies the name of the client.</p> <p>The ref attribute is the variable used for uniquely identifying the client, for example client_id.</p> <p>Both name and ref are optional but at least one of attributes should be used.</p>
Allow (Allow integer)	<p>Specifies the message count to which the Quota will be reset.</p>
Class(Optional)	<p>Refers to the class for which the quota counter will be reset.</p> <pre> &lt;Class name=" {name} " ref="request.header.classIdentifier"&gt; </pre>

Reset Quota policy type defines the following error codes:

Error code	Message
InvalidRLPolicyDefinition	Invalid rate limit policy {0}
NoRLPolicy	Quota policy {0} is not attached.
InvalidCount	Invalid count value {0} for identifier {1} in {2}
FailedToResolveAllowCountRef	Failed to resolve allow count reference {0} for identifier {1} in {2}

## Related Information

[Quota \[page 336\]](#)



## 1.5.1.4.1.19 Service Callout

Use this policy to call an external service from your API flow.

A typical scenario consists of the service callout policy, from the response flow, calling a third party API.

On receiving a response from the backend service, you(API developer) then call the third party API. The response from this API is then appended to the original response to provide a mashed up response to the requesting application.

While using the Service Callout Policy, it is important to understand the other policies that provision to accomplish a task. The Service Callout is usually used with the Assign Message and Extract Variables policy. The Assign Message policy is used to populate the request message sent to the remote service. The Extract Variables policy is used to parse the response and to extract specific content.

A scenario where you use the three policies is as follows:

1. **Assign Message Policy:** Creates a request message, populates HTTP headers, query parameters, sets the HTTP verb, and so on.
2. **Service Callout Policy:** References a message created by the Assign Message policy, defines a target URL for the external call, and defines a name for the response object that the target service returns.
3. **Extract Variables Policy:** Typically defines a JSONPath or XPath expression that parses the message generated by the preceding Service Callout policy. The policy then sets variables containing the values parsed from the Service Callout response.

You can attach the policy in the following locations:

ProxyEndpoint			TargetEndpoint				
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	←Response
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

### Code Syntax

```

1. <!-- This policy will call the url api.exampleAPI.com and put the response
in variable callOutResponse.
-- For examples refer the Flow Variables Table.-->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceCallout async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <Request/>
 <Response>callOutResponse</Response>
 <Timeout>30000</Timeout>
 <HTTPTargetConnection>
 <URL>http://api.exampleAPI.com/API</URL>
 </HTTPTargetConnection>
</ServiceCallout>
2. <!-- This policy will call a dynamic url which is set in the previous step
via policies like javascript or assign variable..
-- For examples refer the Flow Variables Table.-->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceCallout async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <Request/>

```

```

 <Response>callOutResponse</Response>
 <Timeout>30000</Timeout>
 <HTTPTargetConnection>
 <URL>http://{URL_path}</URL>
 </HTTPTargetConnection>
 </ServiceCallout>

```

**Note:** The protocol example http, https can't be set dynamically.

**3. <!-- This policy below refers to an existing API Provider. -->**

**You can use on-premise, and internet type of API Provider in the Service Callout policy.**

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceCallout async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <Request/>
 <Response>callOutResponse</Response>
 <Timeout>30000</Timeout>
 <HTTPTargetConnection>
 <APIProvider>API Provider Name</APIProvider>
 <Path>/sap/opu/odata/iwfind/RMTSAMPLEFLIGHT</Path>
 </HTTPTargetConnection>
</ServiceCallout>

```

**4 (a). <!-- This policy briefs about SSL Info. SSL stands for Secure Socket Layer. It helps in encrypting the link between a web server and a web client, such as a browser or an app.**

**Although the encrypted link ensures that all data passing between the server and the client remains private. The SSL Info does not support if the API Provider is added to HTTPTargetConnection.**

**Henceforth, there can never be a case where API Provider-SSL Configuration conflicts with SSL Info present in the Service Callout Policy.**

**If enabled = "true" in the code;follow the below code. -->**

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceCallout async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <Request/>
 <Response>callOutResponse</Response>
 <Timeout>30000</Timeout>
 <HTTPTargetConnection>
 <URL>https://maps.googleapis.com/maps/api/geocode/json/</URL>
 <SSLInfo>
 <Enabled>true</Enabled>
 <ClientAuthEnabled>true</ClientAuthEnabled>
 <KeyStore>SapKeystore</KeyStore>
 <KeyAlias>SAPKey</KeyAlias>
 <TrustStore>SAPTruststore</TrustStore>
 <Ciphers/>
 <Protocols/>
 </SSLInfo>
 </HTTPTargetConnection>
</ServiceCallout>

```

**4 (b).Use the SSL Info code below, If enabled is set to false.**

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceCallout async="true" continueOnError="false" enabled="false"
xmlns="http://www.sap.com/apimgmt">
 <Request/>
 <Response>callOutResponse</Response>
 <Timeout>30000</Timeout>
 <HTTPTargetConnection>
 <URL>https://maps.googleapis.com/maps/api/geocode/json/</URL>
 </HTTPTargetConnection>
</ServiceCallout>

```

**5. In this Service callout policy, you call a local API Proxy in 2 ways;**

**5 (a). Using local <APIProxy> and <ProxyEndpoint>**

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

```

```

<ServiceCallout async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <Request/>
 <Response>callOutResponse</Response>
 <Timeout>30000</Timeout>
 <LocalTargetConnection>
 <APIProxy>api-admin</APIProxy>
 <ProxyEndpoint>default</ProxyEndpoint>
 </LocalTargetConnection>
</ServiceCallout>

```

In th above code, 'api-admin' is an example for local API Proxy.

**5 (b). Using the path parameter.**

**This can also help in calling a local API Proxy since the path parameter could be a consistent target.**

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceCallout async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <Request/>
 <Response>callOutResponse</Response>
 <Timeout>30000</Timeout>
 <LocalTargetConnection>
 <Path><"API Basepath"></Path>
 </LocalTargetConnection>
</ServiceCallout>

```

**Note:** If you are referring to <LocalTargetConnection> tag, then don't include <HTTPTargetConnection> tag and vice versa.

A callout is typically used with two other policies: Assign Message and Extract Variables.

- Request: Assign Message populates the request message sent to the remote service.
- Response: Extract Variables parses the response and extracts specific content.

Elements and Attributes	Description
Request (Optional)	<p>The variable that contains the request message to be sent by the ServiceCallout.</p> <ul style="list-style-type: none"> <li>• By default, 'clearPayload' is false.</li> <li>• If clearPayload is set to true, the request payload is cleared after the request is sent to the HTTP target.</li> <li>• Use the clearPayload option only if the request message is not required after the Service Callout is executed, because clearPayload allocates memory during message processing.</li> <li>• The policy returns an error if the request message cannot be resolved by the element or it is of an invalid request message type.</li> </ul>

Elements and Attributes	Description
Response (Optional)	<p>Output of the ServiceCallout (usually the response message received from the target) that will be assigned to the response variable.</p> <p>The output generated by the policy is assigned to the variable only when the entire response is read successfully by the policy. If the response message fails for any reason, the policy returns an error.</p> <p>If this element is not specified, the policy execution does not wait for response to be completely read and executes the message flow steps.</p>
Timeout (Optional)	<p>The time in milliseconds that the ServiceCallout policy will wait for a response from the target before exiting. The default timeout for ServiceCallout is determined by the default HTTP timeout setting for API Management, which is 55000 milliseconds (55 seconds).</p>

## Elements and Attributes

## Description

HTTPTargetConnection

Provides transport details such as URL and HTTP properties.

### Note

You can use flow variables to construct the URL in an `HttpTargetConnection` element.

#### `<HTTPTargetConnection>/<URL>` element:

You can supply portion of the URL that changes with a variable. Although, the protocol part of the URL, `http://` beneath, cannot be stated by a variable. In the next example, you use a variable to emphasize the value of query parameter.

```
<URL>http://example.com/forecastrss?w=${request.header.woeid}</URL>
```

Or, set a part of the URL path by a variable:

```
<URL>http://example.com/{request.resourcePath}?w=${request.header.woeid}</URL>
```

If you need to use a variable to quantify the domain and port of the URL, then use one variable alone for the domain and port, and a second variable for the other portion of the URL:

```
<URL>http://{request.dom_port}/{request.resourcePath}</URL>
```

Refer sample code (1)

#### `<LocalTargetConnection>` element:

It quantifies a local proxy within the environment and acts as the target of service callouts.

#### `<LocalTargetConnection>/<ProxyEndpoint>` element

It is a proxy endpoint in the API proxy quantified with the `<APIProxy>` element.

Refer sample code 4 (a)

#### `<LocalTargetConnection>/<Path>` element

It targets the path to the endpoint. The endpoint must refer to a local proxy while making the call.

Refer sample code 4 (b).

## Flow variables

They allow dynamic performance of policies and flows at runtime. It's based on HTTP headers, message content, or Flow context. The following Flow variables are available and predefined after a Service Callout policy is executed.

## Flow Variables Table

Variable	Description
<p>Following is an example of getting Service Callout request and response headers similar to how you would get headers from the main request and response.</p> <pre>calloutResponse.header.HeaderName</pre> <pre>myRequest.header.HeaderName</pre> <p>where <code>calloutResponse</code> is the variable name for the Response in the Service Callout, and <code>myRequest</code> is the variable name for the Request. For example:</p> <pre>calloutResponse.header.Content-Length</pre> <p>returns the Content-Length header of the Service Callout response.</p>	<p>Scope: From the Service Callout forward</p> <p>Type: String</p> <p>Permission: Read/Write</p> <p>A message header in the Service Callout request or response</p>
<pre>servicecallout.requesturi</pre>	<p>Scope: From the Service Callout request forward</p> <p>Type: String</p> <p>Permission: Read/Write</p> <p>The TargetEndpoint URI for a ServiceCallout policy. The URI is the TargetEndpoint URL without the protocol and domain specification.</p>
<pre>servicecallout.{policy-name}.target.url</pre>	<p>Scope: From the Service Callout request forward</p> <p>Type: String</p> <p>Permission: Read/Write</p> <p>The target URL for the Service Callout.</p>
<pre>calloutResponse.content</pre> <p>where <code>calloutResponse</code> is the &lt;Response&gt;variable name in the Service Callout configuration.</p>	<p>Scope: From the Service Callout response forward</p> <p>Type: String</p> <p>Permission: Read/Write</p> <p>The response body from the Service Callout.</p>
<pre>servicecallout.{policy-name}.expectedcn</pre>	<p>Scope: From the Service Callout request forward</p> <p>Type: String</p> <p>Permission: Read/Write</p> <p>The expected Common Name of the TargetEndpoint as referred to in a ServiceCallout policy. This is meaningful only when the TargetEndpoint refers to an TLS/SSL endpoint.</p>

Variable	Description
<code>servicecallout.{policy-name}.failed</code>	<p>Scope: From the Service Callout response forward</p> <p>Type: Boolean</p> <p>Permission: Read/Write</p> <p>Boolean indicating if the policy succeeded, false, or failed, true.</p>

## Errors

This segment defines the fault codes and error messages that are returned.

During the policy execution, the following errors can occur:

Error Code

Error Name	Cause
<code>RequestVariableNotMessageType</code>	The Request variable specified in the policy is not of type Message. For example, if it's a string or other non-message type, you'll see this error.
<code>RequestVariableNotRequest\MessageType</code>	The Request variable specified in the policy is not of type Request Message. For example, if it's a Response type, you'll see this error.
<code>ExecutionFailed</code>	<p>This error can occur when:</p> <ul style="list-style-type: none"> <li>The policy is asked to handle input that is malformed or otherwise invalid.</li> <li>The backend target service returns an error status (by default, 4xx or 5xx).</li> </ul>
<code>ErrorResponseCode</code>	The backend target service returns an error status (by default, 4xx or 5xx).
<code>InvalidExecutionState</code>	JSONThreatProtection[policy_name]: Exceeded string value length at line [line_num]
<code>URLMissing</code>	The <URL> element inside <HTTPTargetConnection> is missing or empty.
<code>ConnectionInfoMissing</code>	This error happens if the policy does not have an <HTTPTargetConnection> element.
<code>InvalidTimeoutValue</code>	This error happens if the <Timeout> value is negative or zero.

## Runtime Errors

Fault Code	Cause
<code>steps.servicecallout.Executionfailed</code>	<p>This error can arise when:</p> <p>The policy is requested to handle input that is distorted or otherwise unacceptable.</p>

Fault Code	Cause
steps.servicecallout.RequestVariableNotMessageType	The Request variable is quantified in the policy and is not of type Message. For instance, if it's a string or other non-message type, you'll encounter this error.
steps.servicecallout.RequestVariableNotRequestMessageType	The Request variable quantified in the policy is not of type Request Message. For instance, if it's a Response type, you'll encounter this error.

### sample error response

#### Sample Code

```
{
 "fault": {
 "detail": {
 "errorCode": "steps.servicecallout.RequestVariableNotMessageType"
 },
 "faultstring": "ServiceCallout[ServiceCalloutGetMockResponse]:
 request variable data_str value is not of type Message"
 }
}
```

### Sample fault rule

#### Sample Code

```
<faultrule name="VariableOfNonMsgType"></faultrule><FaultRule
name="RequestVariableNotMessageType">
 <Step>
 <Name>AM-RequestVariableNotMessageType</Name>
 </Step>
 <Condition>(fault.name = "RequestVariableNotMessageType")</Condition>
</FaultRule>
```

Following fault variables are set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	The [prefix] is servicecallout. [policy_name]: The name of the policy to check.	servicecallout.SC-GetUserData.failed = true
fault.[error_name]	[error_name] = The specific error name to check for as listed in the table above.	fault.name = "RequestVariableNotMessageTypes"

## Related Information

[Assign Message \[page 217\]](#)



## 1.5.1.4.1.20 Spike Arrest

The Spike Arrest policy limits the number of requests forwarded from the point in the processing flow where the policy is attached as a processing step.

You can attach a Spike Arrest policy at the proxy endpoint or the target endpoint. At the proxy endpoint, this policy limits inbound requests. When you attach this policy at the TargetEndpoint, it limits request forwarded to the backend service.

Unlike Quotas, spike arrests are not implemented as counts. Rather, they are implemented as a rate limit which is based on the time the last matching message was processed. If you specify 6 messages per minute, it means that requests can only be submitted at the rate of one per 10 second interval. A second request within 10 seconds on the same API Management server will be rejected. Even with a larger number (200 per second), if two requests come in nearly simultaneously to the same API Management server, one will be rejected. Each successful request will update the spike arrest's last processed count.

No counter is maintained for spike arrests, only a time that the last message was successfully passed through the Spike Arrest policy. On a given API Management server, if a request is received now, all subsequent requests will fail until 10 seconds has elapsed. Since Spike Arrest is not distributed, you might see some discrepancy between the actual behavior of the system and your expected results. In general, you should use Spike Arrest to set a limit that throttles traffic to what your backend services can handle. Do not use Spike Arrest to limit traffic from individual clients.

You can attach the policy in the following locations:

ProxyEndpoint			TargetEndpoint				
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	← Response
	•						
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

### Code Syntax

```
<!-- The policy will limit the number of calls to 30 per minute for a user by
referring to user id in the header -->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<SpikeArrest async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <Identifier ref="request.header.userid"></Identifier>
 <MessageWeight ref="request.header.weight"></MessageWeight>
 <Rate>30pm</Rate>
</SpikeArrest>
```

Elements and Attributes	Description
Identifier ref	Variable used for uniquely identifying the application or client.
MessageWeight ref	Specifies the weight defined for each message.  Message weight is used to modify the impact of a single request on the calculation of the SpikeArrest limit. Message weight can be set by variables based on HTTP headers, query parameters, or message body content. For example, if the SpikeArrest Rate is 10 per minute, and an application submits requests with weight 5, then only 2 messages are permitted per minute from that application.
Rate	Specifies the rate at which to limit the traffic spike (or burst).  Valid value: integer per <min> or <sec> or <variable>.

When a Spike Arrest policy executes, the following Flow variable is populated:

Flow Variable

Variable	Type	Permission	Description
ratelimit.{policy_name}.failed	Boolean	Read-Only	Indicates whether or not the policy failed (true or false).

During the policy execution, the following errors can occur:

Error Code

Error Name	HTTP Status	Cause
SpikeArrestViolation	500	The rate limit is exceeded.
InvalidMessageWeight	500	The message weight value must be an integer.
FailedToResolveSpikeArrestRate	500	The referenced variable used to specify the rate can't be resolved.

Following fault variables are set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	The fault variable [prefix] is ratelimit.  The [policy_name] is the name of the policy that threw the error.	ratelimit.SA-SpikeArrestPolicy.failed = true
fault.[error_name]	[error_name] = The specific error name to check for as listed in the table above.	fault.name Matches "SpikeArrestViolation"

## Related Information

[Quota \[page 336\]](#)

## 1.5.1.4.1.21 OAuth v2.0

OAuth 2.0 defines an authorization protocol for protected API resources.

To ensure that applications are allowed to act on behalf of users, OAuth 2.0 relies on 'access tokens'. To access protected resources, consumer applications must obtain 'access tokens'. The OAuth 2.0 specification defines the various ways that applications can request and use access tokens. API Management provides a policy type that enables you to configure OAuth 2.0 authorization for your APIs.

Setting up OAuth 2.0 authorization for your API is a three step process:

1. Configure a token endpoint: An OAuth token endpoint defines a URI on API Management. The token endpoint is configured with a policy of type OAuthV2. In the OAuthV2 policy, the GenerateAccessToken operation is specified. When this operation is specified, you have the option of configuring one or more grant types. For each grant type specified, an additional set of configuration elements are exposed, providing flexibility in the way that APIs exposed through API Management manage OAuth-based authorization.
2. Apply an OAuth validation policy to protected resource URIs: To enforce OAuth at runtime, attach a policy of type OAuthV2 to a Flow that exposes a protected resource. In the OAuthV2 policy, specify the VerifyAccessToken operation.
3. Configure one or more API products: The VerifyAccessToken operation resolves the access token to an API product for which the application has been approved. The request URI is verified against the list of URIs defined in the API product. If the request URI is included in the list defined by the approved API product, then the request is forwarded to the protected resource.

You can attach this policy in the following locations

ProxyEndpoint			TargetEndpoint				
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	←Response
		•					
					•		
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

:

Element and Attribute Descriptions

Elements & Attributes	Description
AccessToken	By default, VerifyAccessToken expects the access token to be sent in an Authorization header. You can change that default using this element. For example <code>request.queryparam.access_token</code> indicates that the access token should be present as a query parameter.
AppEndUser	This element lets you specify where API Management should look for the end user ID
ClientId	This element lets you specify where API Management should look for the end user ID

Elements & Attributes	Description
Code	This element lets you specify where API Management should look for the authorization code. For example, it could be sent as a query parameter, HTTP header, or form parameter (the default).
ExpiresIn	Enforces the expiry time of access tokens and authorization codes in milliseconds. (For refresh tokens, use <RefreshTokenExpiresIn>.) The expiry time value is a system generated value plus the <ExpiresIn> value. If <ExpiresIn> is set to -1, the token or code is given an infinite lifetime. If <ExpiresIn> is not specified, the system applies a default value configured at the system level.
ExternalAccessToken	Tells API Management where to find an external access token
GenerateResponse	If set to true, the policy generates and returns a response
GenerateErrorResponse	If set to true, the policy generates and returns a response if the ContinueOnError attribute is set to true. If false (the default), no response is sent.
GrantType	Tells the policy where to find the grant type parameter that is passed in a request.
RedirectUri	Specifies where to should look for the <code>redirect_uri</code> parameter in the request.
RefreshToken	When requesting an access token using a refresh token, you must supply the refresh token in the request. This element lets you specify where API Management should look for the refresh token. For example, it could be sent as a query parameter, HTTP header, or form parameter
RefreshTokenExpiresIn	Enforces the expiry time of refresh tokens in milliseconds. The expiry time value is a system generated value plus the <RefreshTokenExpiresIn> value. If <RefreshTokenExpiresIn> is set to -1, the refresh token is given an infinite lifetime. If <RefreshTokenExpiresIn> is not specified, the system applies a default value configured at the system level.
ResponseType	This element informs API Management which grant type the client app is requesting. It is used only with the authorization code and implicit grant type flows.
ReuseRefreshToken	When set to true, the existing refresh token is reused until it expires. If false, a new refresh token is issued by API Management when a valid refresh token is presented.
PassWord	This element is used with the password grant type only. With the password grant type, user credentials (password and username) must be made available to the OAuthV2 policy. The <PassWord> and <UserName> elements are used to specify variables where API Management can find these values. If these elements are not specified, the policy expects to find the values (by default) in form parameters named username and password.
Scope	If this element is present in one of the GenerateAccessToken or GenerateAuthorizationCode policies, it is used to specify which scopes to grant the token or code.
State	In cases where the client app must send the state information to the authorization server, this element lets you specify where API Management should look for the state values. For example, it could be sent as a query parameter or in an HTTP header.

Elements & Attributes	Description
StoreToken	Set this element to true when the <ExternalAuthorization> element is true. The <StoreToken> element tells API Management to store the external access token. Otherwise, it will not be persisted.

Following flow variables are populated when the policy is executed:

Flow Variables

Variable	Description
organization_name	The name of the organization where the proxy is executing.
developer.id	The ID of the developer associated with the registered client app.
developer.app.name	The name of the developer associated with the registered client app.
client_id	The client ID of the registered client app.
grant_type	The grant type associated with the request.
token_type	The token type associated with the request.
access_token	The access token that is being verified.
accesstoken.{custom_attribute}	A named custom attribute in the access token.
issued_at	The date the access token was issued.
expires_in	The expiration time for the access token
status	The status of the access token (for example, approved or revoked).
scope	The scope (if any) associated with the access token
apiproduct.<custom_attribute_name>	A named custom attribute of the API product associated with the registered client app.
apiproduct.name	The name of the API product associated with the registered client app.

Variable	Description
App-specific variables	
app.name	
app.id	
app.accessType	
app.callbackUrl	
app.status	
app.scopes	
app.appFamily	
app.apiproducts	
app.appParentStatus	
app.appType	
app.appParentId	
app.created_by	
app.created_at	
app.last_modified_at	
app.last_modified_by	
app.{custom_attributes}	
Developer-specific variables	Note : If the app.appType is "Developer", then developer attributes are populated.
developer.id	
developer.userName	
developer.firstName	
developer.lastName	
developer.email	
developer.status	
developer.apps	
developer.created_by	
developer.created_at	
developer.last_modified_at	
developer.last_modified_by	
developer.{custom_attributes}	

During the policy execution, the following errors can occur:

## Error Cause

Error Name	Cause
InvalidClientIdentifier	The client identifier sent from the client is invalid or missing. Check to be sure you are using the correct client key and secret values for the Developer App associated with your proxy. Typically, these values are sent as a Base64 encoded Basic Authorization header.  Important: This error name used to be called <code>invalid_client</code> .
<code>invalid_client</code>	This error name is no longer used. It was replaced by <code>InvalidClientIdentifier</code> .
<code>invalid_request</code>	This error name is used for multiple different kinds of errors, typically for missing or incorrect parameters sent in the request. If <code>&lt;GenerateResponse&gt;</code> is set to <code>false</code> , use fault variables (described below) to retrieve details about the error, such as the fault name and cause.
<code>invalid_access_token</code>	The access token sent from the client is invalid.
<code>FailedToResolveToken</code>	The policy expected to find a token in a variable specified in the <code>&lt;Tokens&gt;</code> element, but the variable could not be resolved.
<code>FailedToResolveClientId</code>	The policy expected to find the Client ID in a variable specified in the <code>&lt;ClientId&gt;</code> element, but the variable could not be resolved.
<code>FailedToResolveAccessToken</code>	The policy expected to find an access token in a variable specified in the <code>&lt;AccessToken&gt;</code> element, but the variable could not be resolved.
<code>FailedToResolveRefreshToken</code>	The policy expected to find a refresh token in a variable specified in the <code>&lt;RefreshToken&gt;</code> element, but the variable could not be resolved.
<code>FailedToResolveAuthorizationCode</code>	The policy expected to find an authorization code in a variable specified in the <code>&lt;Code&gt;</code> element, but the variable could not be resolved.
<code>UnSupportedGrantType</code>	The client specified a grant type that is unsupported by the policy
<code>InvalidTokenType</code>	The <code>&lt;Tokens&gt;/&lt;Token&gt;</code> element requires you to specify the token type (for example, <code>refresh_token</code> ). If the client passes the wrong type, this error is thrown.
<code>InvalidAPICallAsNoApiProductMatchFound</code>	The API proxy is not in the Product associated with the access token.
<code>InsufficientScope</code>	The access token presented in the request has a scope that does not match the scope specified in the verify access token policy.
<code>InvalidParameter</code>	The policy must specify either an access token or an authorization code, but not both.
<code>MissingParameter</code>	The response type is <code>token</code> , but no grant types are specified.
<code>InvalidValueForExpiresIn</code>	For the <code>&lt;ExpiresIn&gt;</code> element, valid values are positive integers and <code>-1</code> .
<code>InvalidValueForRefreshTokenExpiresIn</code>	For the <code>&lt;RefreshTokenExpiresIn&gt;</code> element, valid values are positive integers and <code>-1</code> .
<code>InvalidGrantType</code>	An invalid grant type is specified in the <code>&lt;SupportedGrantTypes&gt;</code> element.
<code>ExpiresInNotApplicableForOperation</code>	Be sure that the operations specified in the <code>&lt;Operations&gt;</code> element support expiration.
<code>RefreshTokenExpiresInNotApplicableForOperation</code>	Be sure that the operations specified in the <code>&lt;Operations&gt;</code> element support refresh token expiration.

Error Name	Cause
GrantTypesNotApplicableForOperation	Be sure that the grant types specified in <SupportedGrantTypes> are supported for the specified operation.
OperationRequired	You must specify an operation in this policy using the <Operation> element.
InvalidOperation	You must specify a valid operation in this policy using the <Operation> element.
TokenValueRequired	You must specify a token <Token> value in the <Tokens> element.

Following fault variables are set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	The [prefix] is oauthV2. The [policy_name] is the name of the policy that threw the error.	oauthV2.GenerateAccessToken.failed = true
[prefix].[policy_name].fault.name	The [prefix] is oauthV2. The [policy_name] is the name of the policy that threw the error.	oauthV2.GenerateAccessToken.fault.name = invalid_request  Note: For the VerifyAccessToken operation, the fault name includes this suffix: keymanagement.service For example: keymanagement.service.invalid_access_token
[prefix].[policy_name].fault.cause	The [prefix] is oauthV2. The [policy_name] is the name of the policy that threw the error.	oauthV2.GenerateAccessToken.cause = Required param : grant_type
fault.name = [error_name]	[error_name] is the specific error name to check for as listed in the table above.	fault.name = "invalid_request"

## Related Information

[Generate Access Token \[page 361\]](#)

[Generate Authorization Code \[page 362\]](#)

[Verify Access Tokens \[page 363\]](#)

[Designing OAuth v2.0 Policies \[page 364\]](#)

[OAuth 2.0 Grant Types \[page 367\]](#)



## 1.5.1.4.1.21.1 Generate Access Token

OAuth 2.0 policies are used both to generate and to validate OAuth 2.0-compliant tokens. To generate tokens on behalf of application end users, OAuth 2.0 policies that specify the `GenerateAccessToken` operation are attached to a token endpoint.

### Note

Deploy a single API proxy to function as a token endpoint for all API proxies in an environment. A single API proxy configured as a token endpoint can support multiple grant types. By setting up a single token endpoint, you can publish a unified set of URIs that application developers can use to obtain tokens.

A token endpoint is simply a URI path that the system uses to identify requests for access tokens. On API Management, a token endpoint is a conditional flow to which an OAuthV2 policy is attached. The OAuthV2 policy specifies the `GenerateAccessToken` operation as an element. For example, to configure a token endpoint that generates tokens on requests to the URI path `/accesstoken`:

### Sample Code

```
<Flow name="TokenEndpoint">
 <Condition>proxy.pathsuffix MatchesPath "/accesstoken"</Condition>
 <Request>
 <Step><Name>GenerateAccessToken</Name></Step>
 </Request>
</Flow>
```

### Note

An example payload for the policy is as follows.

### Code Syntax

```
<OAuthV2 async="false" continueOnError="false" enabled="true" xmlns="http://
www.sap.com/apimgmt">
 <!-- this flag has to be set when you want to work with third-party access
 tokens -->
 <ExpiresIn ref="kvm.expiry.value">360000</ExpiresIn> <!-- in mili seconds -->
 <ExternalAuthorization>false</ExternalAuthorization>
 <GrantType>request.queryparam.grant_type</GrantType>
 <Operation>GenerateAccessToken</Operation>
 <PassWord>request.formparam.password</PassWord>
 <RedirectUri/>
 <RefreshToken/>
 <RefreshTokenExpiresIn ref="kvm.expiry.value">240000</RefreshTokenExpiresIn>
 <GenerateResponse enabled="true"/>
 <SupportedGrantTypes>
 <GrantType>client_credentials</GrantType>
 </SupportedGrantTypes>
</OAuthV2>
```

These variables are set when the `GenerateAccessToken` policy operation executes successfully for the authorization code, password, and client credentials grant type flows. Note that refresh token variables do not apply to and are not set by the client credentials grant type flow.

Access Token

Variable	Description
<code>oauthv2accesstoken.{policy_name}.token_type</code>	Will be set to accesstoken.
<code>oauthv2accesstoken.{policy_name}.expires_in</code>	The expiry value for the token.
<code>oauthv2accesstoken.{policy_name}.refresh_token</code>	The refresh token generated when the policy executes.
<code>oauthv2accesstoken.{policy_name}.refresh_token_expires_in</code>	The lifespan of the refresh token, in seconds.
<code>oauthv2accesstoken.{policy_name}.refresh_token_issued_at</code>	This time value is the string representation of the corresponding 32-bit timestamp quantity.
<code>oauthv2accesstoken.{policy_name}.refresh_token_status</code>	Set to approved or revoked.
<code>oauthv2accesstoken.{policy_name}.scope</code>	List of available OAuth scopes.

For information on the various `GrantType` supported, see [OAuth 2.0 Grant Types \[page 367\]](#). For information on the various field descriptions (supported elements and attributes), see [Designing OAuth v2.0 Policies \[page 364\]](#).

## Related Information

[Designing OAuth v2.0 Policies \[page 364\]](#)

[OAuth 2.0 Grant Types \[page 367\]](#)

### 1.5.1.4.1.21.2 Generate Authorization Code

Similarly, you can configure authorization endpoints to issue authorization codes.

For example:

#### Sample Code

```
<Flow name="AuthorizationEndpoint">
 <Condition>proxy.pathsuffix == "/authorize"</Condition>
 <Request>
 <Step><Name>GenerateAuthCode</Name></Step>
 </Request>
</Flow>
```

This policy needs to be attached to the response to generate the authorization code.

## Note

An example payload for the policy is as follows.

## Code Syntax

```
<OAuthV2 async="false" continueOnError="false" enabled="true" xmlns="http://www.sap.com/apimgmt">
 <ClientId>request.queryparam.client_id</ClientId>
 <Operation>GenerateAuthorizationCode</Operation>
 <RedirectUri>request.queryparam.redirect_uri</RedirectUri>
 <GenerateResponse enabled="true"/>
 <ResponseType>request.queryparam.response_type</ResponseType>
 <Scope>request.queryparam.scope</Scope>
 <Tokens/>
</OAuthV2>
<!-- sample API call to get the auth code -->
https://<auth_endpoint>?redirect_uri=https://<your-app-redirect-url>
&response_type=code&scope=read&state=1&client_id=<a_valid_app_key>
Here the response will be HTTP 302 and the code will be sent to app-redirect-
url as a query parameter e.g.
HTTP 302 https://<your-app-redirect-url>?code=<the_generated_code>
&state=1&scope=read
```

These variables are set when the GenerateAuthorizationCode policy operation executes successfully:

Authorization Code

Variable	Description
oauthv2authcode.{policy_name}.code	The authorization code generated when the policy executes.
oauthv2authcode.{policy_name}.redirect_uri	The redirect URI associated with the registered client app.
oauthv2authcode.{policy_name}.scope	The optional OAuth scope passed in the client request.
oauthv2authcode.{policy_name}.client_id	The client ID passed in the client request.

For information on the various GrantType supported, see [OAuth 2.0 Grant Types \[page 367\]](#). For information on the various field descriptions (supported elements and attributes), see [Designing OAuth v2.0 Policies \[page 364\]](#).

## 1.5.1.4.1.21.3 Verify Access Tokens

Once a token endpoint is set up for an API proxy, a corresponding OAuthV2 policy that specifies theVerifyAccessToken operation is attached to the Flow that exposes the protected resource.

## Example

To ensure that all requests to an API are authorized, the following policy enforces access token verification:

### Sample Code

```
<OAuthV2>
 <Operation>VerifyAccessToken</Operation>
</OAuthV2>
```

The policy is attached to the API resource to be protected. To ensure that all requests to an API are verified, attach the policy to the proxy endpoint request PreFlow, as follows:

### Sample Code

```
<PreFlow>
 <Request>
 <Step><Name>VerifyOAuthAccessToken</Name></Step>
 </Request>
</PreFlow>
```

The following optional elements can be used to override the default settings for the VerifyAccessToken operation

Name	Description
Scope	<p>A space separated list of scopes that must be present in the access token for verification to succeed. For example, the following policy will check the access token to ensure that it contains the scopes READ and WRITE:</p> <pre>Sample Code</pre> <pre>&lt;OAuthV2&gt;   &lt;Operation&gt;VerifyAccessToken&lt;/ Operation&gt;   &lt;Scope&gt;READ WRITE&lt;/Scope&gt; &lt;/OAuthV2&gt;</pre>
AccessToken	<p>The variable where the access token is expected to be located. For example, request.queryparam.accesstoken. By default, the access token is expected to be presented by the application in the Authorization HTTP header, according to the OAuth 2.0 specification. Use this setting if the access token is expected to be presented in a nonstandard location. Such location may be a query parameter, or an HTTP header with a name other than Authorization.</p>

For information on the various field descriptions (supported elements and attributes), see [Designing OAuth v2.0 Policies \[page 364\]](#).

## 1.5.1.4.1.21.4 Designing OAuth v2.0 Policies

The table below illustrates the various elements and attributes used in the OAuth policies:

			Related Operation and Grant Type
Name	Description	Valid Values	Combinations
Operation	The OAuth 2.0 operation implemented by the policy	GenerateAccessToken GenerateAccessTokenImplicitGrant GenerateAuthorizationCode RefreshAccessToken VerifyAccessToken	All
ExpiresIn(optional)	ExpiresIn enforces the expiry time of the authorization code in milliseconds. The expiry time of authorization code is system generated plusExpiresInvalue. IfExpiresInis -1, the system considers it as an infinite life time. If it is not specified, the system applies a default value configured at system level.	GenerateAccessToken GenerateAccessTokenImplicitGrant GenerateAuthorizationCode RefreshAccessToken VerifyAccessToken	All
SupportedGrant Types	Specifies the GrantTypes supported by a token endpoint.  An endpoint may support multiple GrantTypes (that is, a single endpoint can be configured to distribute access tokens for multiple GrantTypes.)	client_credentials authorization_code implicit	All
Code	The expected location in the request message where the authorization code must be presented to the token endpoint	Any variable setting. For example request.queryparam.auth_code indicates that the authorization code should be present as a query parameter, as, for example, ?auth_code=AfGlvs9. To require the authorization code in an HTTP header, for example, set this value to request.header.auth_code.	GenerateAccessToken with grant typeauthorization_code

Name	Description	Valid Values	Related Operation and Grant Type Combinations
ClientId	The expected location in the request message where the client_id (the app's consumer key) must be presented to the token endpoint	<p>Any variable setting. For example request.queryparam.client_id indicates that the client_id should be present as a query parameter, as, for example, ?client_id=AfGlvs9.</p> <p>To require the ClientId in an HTTP header, for example, set this value</p> <p>to request.header.client_id.</p>	<p>GenerateAccessToken</p> <p>Implicit: Optional</p> <p>GenerateAuthorization</p> <p>Code:Optional</p>
RedirectUri	The expected location in the request message where the RedirectUri must be presented to the token endpoint.	<p>Any variable setting. For example, request.queryparam.redirect_uri indicates that the RedirectUri should be present as a query parameter, as, for example, ?redirect_uri=login.myapp.com. To require the RedirectUri in an HTTP header, for example, set this value to request.header.redirect_uri.</p>	<p>GenerateAccessToken</p> <p>Implicit: Optional</p> <p>GenerateAuthorization</p> <p>Code:Optional</p>
Scope	The expected location in the request message where the scope must be presented to the token endpoint.	<p>Any variable setting. For example, request.queryparam.scope indicates that the scope should be present as a query parameter, as, for example, ?scope=READ. To require the scope in an HTTP header, for example, set this value to request.header.scope.</p>	All: Optional
State	The expected location in the request message where the state must be presented to the token endpoint.	<p>Any variable setting. For example request.queryparam.state indicates that the state should be present as a query parameter, as, for example, ?state=HjoiuKJH32.</p> <p>To require the state in an HTTP header, for example, set this value to request.header.state.</p>	<p>authorization_code,</p> <p>password</p>

Name	Description	Valid Values	Related Operation and Grant Type Combinations
AppEndUser	The expected location in the request message where the state must be presented to the token endpoint.	Any variable setting. For example request.queryparam.app_enduserindicates that the AppEndUser should be present as a query parameter, as, for example, ?app_enduser=ntesla@theramin.com. To require the AppEndUser in an HTTP header, for example, set this value to request.header.app_enduser.	All: Optional
UserName	The expected location in the request message where the UserName must be presented to the token endpoint.	Any variable setting. For example request.queryparam.username indicates that the username should be present as a query parameter, as, for example, ?username=joe. To require the UserName in an HTTP header, for example, set this value to request.header.username.	All: Optional
Password	The expected location in the request message where the Password must be presented to the token endpoint.	Any variable setting. For example request.queryparam.password indicates that the Password should be present as a query parameter, as, for example, ?password=changeit. To require the Password in an HTTP header, for example, set this value to request.header.password.	All: Optional
GenerateResponse	An element used in token endpoints, authorization endpoints, and refresh endpoints to indicate that a response should be generated for requests, and that no further processing should take place. (Indicates that the policy is an endpoint.)	N/A	All: Optional

## 1.5.1.4.1.21.5 OAuth 2.0 Grant Types

OAuth 2.0 policies such as generate access token and generate authorization code use the GrantType method element. The below table illustrates the three supported grant types:

Grant Type	Description
Clientcredentials	"Two-legged" OAuth, usually implemented for trusted clients (for example, applications developed by the API provider themselves).
Authorizationcode	"Three-legged" OAuth, which enables the application end users to obtain an access token without exposing credentials to the application. The application requests an access token using an authorization code returned by the intermediary who authenticates the application end user. API Platform can act as both authorization server (generating authorization codes) and as a token endpoint (issuing access tokens in return for valid authorization codes).
Implicit	A variation on authorization code, usually enforced for browser-based applications that are implemented in scripting languages such as JavaScript

## 1.5.1.4.1.22 OAuth v2.0 GET

API Management generates and manages a set of OAuth resources for apps.

Depending on the OAuth configuration for an organization, API Management will generate and manage access tokens, authorization codes, and refresh tokens. For each OAuth resource that it generates, API Management also creates and stores a profile.

The GetOauthV2Info policy type enables you to get attributes of type tokens and authorization codes and to make them available to policies and code executing in an API proxy. This policy type can be useful when you need to configure dynamic, conditional behavior based on a value in an access token.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			← Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An access token has the following JSON representation on API Management:

### Code Syntax

```
{
 "issued_at" : "1847470170943",
 "application_name" : "efd1903j-b667-4431-cf82-bbb3abf9t586",
 "scope": "READ",
 "status" : "approved",
 "api_product_list" : "[FreeProduct]",
 "expires_in" : "2450",
 "developer_email" : "adam@sap.com",
 "organization_id" : "0",
 "refresh_token" : "64XMxgDyRFpFyXOaApj1N7AGIPnN2Ize",
 "client_id" : "ceGYedE0Y9Z0T35PEMaAXYphBJCGdrND",
 "access_token" : "shTUmelIgeSKin0TODcGLXBNe9vp",
 "organization_name" : "apifactory",
 "refresh_count" : "0"
}
```



The properties of an access token profile are set as variables whenever a token is generated or validated. Sometimes, however, you will need to access these properties when no token generation or validation occurs. To do so, you can explicitly populate the access token profile by using the GetOAuthV2Info policy.

An example payload for the policy is as follows:

```

{ } Code Syntax

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<GetOAuthV2Info async="false" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <AccessToken ref="request.access_token"></AccessToken>
 <ClientId ref="request.header.client_id"></ClientId>
</GetOAuthV2Info

```

OAuth v2.0 GET policy defines the following elements:

Field Name	Description
AccessToken (Optional)	Use this element to retrieve the profile for an OAuth 2.0 access token.
AuthorizationCode (Optional)	Use this element to retrieve the profile for an OAuth 2.0 authorization code.
ClientId	Use this element to retrieve information about ClientId.
RefreshToken (Optional)	Use this element to retrieve the profile for an OAuth 2.0 refresh token.

OAuth v2.0 GET policy defines the following errors:

Error Name	Cause
invalid_access_token	The access token sent to the policy is invalid.
expired_access_token	The access token sent to the policy is expired.
invalid_refresh_token	The refresh token sent to the policy is invalid.
refresh_token_expired	The refresh token sent to the policy is expired.
invalid_client-invalid_client_id	The client ID sent to the policy is invalid.
invalid_request-authorization_code_invalid	The authorization code sent to the policy is invalid.
authorization_code_expired	The authorization code sent to the policy is expired.

Following flow variables are populated and is used in cases where you need the profile data:

## Flow Variables

Variable Type	When	Variables list
Client ID variables	These variables are populated when the <ClientId> operation executes	oauthv2client.{policy_name}.client_id oauthv2client.{policy_name}.client_secret oauthv2client.{policy_name}.redirection_uris oauthv2client.{policy_name}.developer.email oauthv2client.{policy_name}.developer.app.name oauthv2client.{policy_name}.developer.id oauthv2client.{policy_name}.{developer_app_custom_attribute_name}
Access token variables	These variables are populated when the <AccessToken> operation executes	oauthv2accesstoken.{policy_name}.access_token oauthv2accesstoken.{policy_name}.scope oauthv2accesstoken.{policy_name}.refresh_token oauthv2accesstoken.{policy_name}.accesstoken.{custom_attribute_name} oauthv2accesstoken.{policy_name}.developer.id oauthv2accesstoken.{policy_name}.developer.app.name oauthv2accesstoken.{policy_name}.expires_in oauthv2accesstoken.{policy_name}.status
Authorization code variables	These variables are populated when the <AuthorizationCode> operation executes	oauthv2authcode.{policy_name}.client_id oauthv2authcode.{policy_name}.organization_id oauthv2authcode.{policy_name}.issued_at oauthv2authcode.{policy_name}.expires_in oauthv2authcode.{policy_name}.redirect_uri oauthv2authcode.{policy_name}.status oauthv2authcode.{policy_name}.state oauthv2authcode.{policy_name}.scope oauthv2authcode.{policy_name}.id oauthv2authcode.{policy_name}.{auth_code_custom_attribute_name}

Variable Type	When	Variables list
Refresh token variables	These variables are populated when the <RefreshToken> operation executes	oauth2refresh_token.{policy_name}.access_token oauth2refresh_token.{policy_name}.refresh_token oauth2refresh_token.{policy_name}.client_id oauth2refresh_token.{policy_name}.refresh_count oauth2refresh_token.{policy_name}.organization_name oauth2refresh_token.{policy_name}.refresh_token_expires_in oauth2refresh_token.{policy_name}.refresh_token_issued_at oauth2refresh_token.{policy_name}.refresh_token_status oauth2refresh_token.{policy_name}.developer.email oauth2refresh_token.{policy_name}.developer.id oauth2refresh_token.{policy_name}.developer.app.name oauth2refresh_token.{policy_name}.developer.app.id oauth2refresh_token.{policy_name}.access_token.{custom_attribute_name}

### 1.5.1.4.1.23 OAuth v2.0 SET

API Management generates and distributes OAuth access tokens to apps. API Management allows you to add or update custom attributes associated with an access token. This policy cannot be used to change fields like scope, status, expires\_in, developer\_email, client\_id, org\_name, or refresh\_count. If an attribute already exists, this policy updates it. If it does not exist, the policy adds it. The access token referenced must be valid and in an approved state.

When API Management generates these OAuth artifacts, it also generates 'profile' that contains metadata related to the token or code. For example, the default access token profile contains name/value pairs that define expiration time, the associated app and developer, and so on.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

The JSON representation of an API Management access token looks like the following:

#### Code Syntax

```
{
 "issued_at" : "1847470170943",
 "application_name" : "efd1903j-b667-4431-cf82-bbb3abf9t586",
 "scope": "READ",
 "status" : "approved",
 "api_product_list" : "[FreeProduct]",
 "expires_in" : "2450",
 "developer.email" : "adam@sap.com",
 "organization_id" : "0",
 "refresh_token" : "64XMXgDyRFpFyXOaApj1N7AGIPnN2Ize",
 "client_id" : "ceGYedEOY9Z0T35PEMaAXYphBJCGdrND",
 "access_token" : "shTUmeIlgeSKin0TODcGLXBNe9vp",
 "organization_name" : "apifactory",
 "refresh_count" : "0"
}
```

In some situations, you will need to update the profile of an access token. For example, you may want to embed a tag that is unique to your business. You might need to embed a department name, a customer ID or more technically, a session identifier, in the access token.

There are two ways to do this: Using an API call or using the SetOAuthV2Info policy. You can call the management API to directly update the access token's profile.

Use the policy when you need tokens to be updated at runtime, such as at the time when the token or code is generated by API Management.

An example payload for the policy is as follows:

#### Code Syntax

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<SetOAuthV2Info async="false" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <AccessToken ref="request.access_token"></AccessToken>
 <Attributes>
 <Attribute display="true" name="department.id">marketing</Attribute>
 <Attribute display="true" name="scope">READ, WRITE</Attribute>
 </Attributes>
</SetOAuthV2Info>
```

OAuth v2.0 SET policy defines the following elements:

Field Name	Description
AccessToken	Use the ref attribute to identify the variable where the access token is located. For example, if the access token is attached to request message as a query parameter, specify request.queryparam.access_token.
Attributes	A set of attributes in the access token profile that will be modified or augmented.

Field Name	Description
Attribute	<p>An individual attribute to update.</p> <p>The name attribute identifies the property of the access token profile to be updated. For example, to modify the access token's scope property, specify scope as the value of the name attribute.</p> <p>The ref attribute specifies either variable or a static setting whose value will be used as the value of the access token profile property that will be updated. For example to update the attribute scope with the value READ, WRITE:</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p><b>Sample Code</b></p> <pre>&lt;Attribute name="scope" ref=""&gt;READ,WRITE&lt;/Attribute&gt;</pre> </div>

On success, the following flow variables is set:

- `oauthv2accesstoken.{policyName}.access_token`
- `oauthv2accesstoken.{policyName}.client_id`
- `oauthv2accesstoken.{policyName}.refresh_count`
- `oauthv2accesstoken.{policyName}.organization_name`
- `oauthv2accesstoken.{policyName}.expires_in`
- `oauthv2accesstoken.{policyName}.refresh_token_expires_in`
- `oauthv2accesstoken.{policyName}.issued_at`
- `oauthv2accesstoken.{policyName}.status`
- `oauthv2accesstoken.{policyName}.api_product_list`
- `oauthv2accesstoken.{policyName}.token_type`
- `oauthv2accesstoken.{policyName}.{custom_attribute_name}`

OAuth v2.0 SET policy defines the following errors:

Error Name	Cause
<code>invalid_access_token</code>	The access token sent to the policy is invalid.
<code>expired_access_token</code>	The access token sent to the policy is expired.

## 1.5.1.4.1.24 Python Script

This policy is used to configure the Python Script code to execute within the context of an API proxy.

The Python Script policy allows you to add a custom-built python functionality to your API proxy flow, wherein the functionality you need isn't supported through the existing policies available in API Management.

Jython version 2.5.2 provides the required python language support. You can find the Jython version 2.5.2 libraries in the following link:

<https://www.jython.org/jython-old-sites/docs/index.html>

### Note

The third-party libraries you add must be implemented in pure python language only.

A Python policy contains no actual code. Instead, a Python policy references a Python 'resource' and defines the Step in the API flow where the Python script executes. The Python Script resource must always have the .py extension.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

### Code Syntax

```
<!-- This sample Python Script policy demonstrates how python scripts are
executed. -->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Script timeLimit="200" async="false" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <IncludeURL>py://dependency_script.py</IncludeURL>
 <ResourceURL>py://mainscript.py</ResourceURL>
</Script>
```

An example of a mainscript.py script

### Code Syntax

```
This is a sample mainscript.py
x = 1
if x == 1:
 # indented four spaces
 print "x is 1."
```

Attribute Name	Description	Required
timeLimit	Specifies the maximum time (in milliseconds) that the script is permitted to execute.	Yes

Attribute Name	Description	Required
IncludeURL	<p>This attribute specifies the python file to be loaded as dependency to the primary python file specified within the ResourceURL attribute. You can store the dependency python file under <code>APIProxy/FileResources/</code> in the API proxy bundle.</p> <p>The name of the dependency python file must be of type 'String'.</p> <div style="border: 1px solid #0070c0; padding: 5px; margin-top: 10px;"> <p><b>Note</b></p> <p>The python libraries you add must be implemented using pure python language only.</p> </div> <p>You can include multiple dependency python files with additional IncludeURL attributes. The scripts are evaluated in the order in which they are listed in the policy.</p>	Optional
ResourceURL	<p>This attribute specifies the primary python file that executes in the API flow. You must store this python file under <code>/APIProxy/FileResources/</code> in the API proxy bundle.</p> <p>The name of the primary python file must be of type 'String'.</p> <div style="border: 1px solid #0070c0; padding: 5px; margin-top: 10px;"> <p><b>Note</b></p> <p>The python libraries you add must be implemented using pure python language only.</p> </div>	Yes

During the policy execution, the following errors can occur:

Attribute Name	Description
InvalidResourceUrlFormat	This error occurs when the format of the resource URL specified within the <code>&lt;ResourceURL&gt;</code> or the <code>&lt;IncludeURL&gt;</code> attribute of the Python Script policy is invalid, resulting in the failure of API proxy deployment.
InvalidResourceUrlReference	This error occurs when the <code>&lt;ResourceURL&gt;</code> or the <code>&lt;IncludeURL&gt;</code> attributes refer to a python file that doesn't exist, resulting in failure of API proxy deployment.
NoResourceForURL	The <code>&lt;ResourceURL&gt;</code> and <code>&lt;IncludeURL&gt;</code> attributes refer to a Python Script file that doesn't exist.

### Note

For added security API Management python runtime executes in a restricted mode, where import is not allowed for `os`, `sys`, and `java.lang`.

## 1.5.1.4.1.25 SAML Assertion Policy

The Security Assertion Markup Language (SAML) Assertion policy enables API proxies to validate and generate SAML assertions in inbound and outbound requests, respectively.

SAML specification defines formats and protocols that enable applications to exchange XML-formatted information for authentication and authorization. A "security assertion" is a trusted token that describes an attribute of an app, an app user, or some other participant in a transaction. Security assertions are managed and consumed by two types of entities:

- Identity providers: Generate security assertions on behalf of participants.
- Service providers: Validate security assertions through trusted relationships with identity providers.

The API platform can act as an identity provider and as a service provider. It acts as an identity provider by generating assertions and attaching them to request messages, making those assertions available for processing by backend services. It acts as a service provider by validating assertions on inbound request messages.

### Generate SAML Assertion

Policy Processing:

- If the message type is not XML, and IgnoreContentType is not set to true, raise a fault.
- If the Template is set, process the template as described for the AssignMessage policy. If any variables are missing and IgnoreUnresolvedVariables is not set, raise a fault.  
If the Template is not set, construct an assertion that includes the values of the Subject and Issuer parameters or their references.
- Sign the assertion using the specified key.
- Add the assertion to the message at the specified XPath.

Sample Schema for SAML Assertion Generation

#### Sample Code

```
<!-- The policy will generate saml assertion and assign assertion to the
varibale defined in xpath-->
<GenerateSAMLAssertion async="false" continueOnError="false" enabled="true"
ignoreContentType="false" xmlns="http://www.sap.com/apimgmt">
 <Issuer ref="saml.issuer">Issuer name</Issuer>
 <KeyStore>
 <Name >saml_key_store</Name>
 <Alias >key_store</Alias>
 </KeyStore>
 <OutputVariable>
 <FlowVariable>sapapim.assertion</FlowVariable>
 <Message name="request">
 <Namespaces>
 <Namespace prefix="env">http://schemas.xmlsoap.org/soap/envelope/</
Namespace>
 </Namespaces>
 <XPath>/env:Envelope/env:Header</XPath>
 </Message>
 </OutputVariable>
 <Subject ref="saml.subject">Subject name</Subject>
 <Template ignoreUnresolvedVariables="false"><![CDATA[
<saml2:Assertion ID="{saml.id}" IssueInstant="{saml.issueInstant}"
Version="2.0" xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
```



```

 <saml2:Issuer
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">{saml.issuer}</
saml2:Issuer>
 <saml2:Subject xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
 <saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:unspecified">{saml.subject}</saml2:NameID>
 <saml2:SubjectConfirmation
Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
 <saml2:SubjectConfirmationData
NotOnOrAfter="{sapapim.notOnorAfter}" Recipient="{saml.recipient}"/>
 </saml2:SubjectConfirmation>
 </saml2:Subject>
 <saml2:Conditions
NotBefore="{sapapim.notBefore}" NotOnOrAfter="{sapapim.notOnorAfter}"
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
 <saml2:AudienceRestriction>
 <saml2:Audience>{saml.audience}</saml2:Audience>
 </saml2:AudienceRestriction>
 </saml2:Conditions>
 <saml2:AuthnStatement AuthnInstant="{sapapim.timestamp}"
SessionNotOnOrAfter="{sapapim.notOnorAfter}"
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
 <saml2:AuthnContext>
 <saml2:AuthnContextClassRef>urn:none</saml2:AuthnContextClassRef>
 </saml2:AuthnContext>
 </saml2:AuthnStatement>
</saml2:Assertion>
]]></Template>
</GenerateSAMLAssertion>

```

## Element Reference

Elements and Attributes	Description
Name	The name of the policy instance. The name must be unique in the organization. Characters you can use in the name are restricted to: A-Z0-9._\-\$ %. However, the Management UI enforces additional restrictions, such as automatically removing characters that are not alphanumeric.
ignoreContentTypeattribute	A boolean that needs to be set to false. By default, the assertion will not be generated if the content type of the message is not an XML Content-Type. If this is set to true, then the message will be treated as XML regardless of the Content-type.
Issuer	The unique identifier of the identity provider. If the optional refattribute is present, then the value of Issuer will be assigned at runtime based on the specified variable. If the optional refattribute is not present, then the value of Issuer will be used.
KeyStore	The name of the KeyStore that contains the private key and the alias of the private key used to digitally sign SAML assertions.
OutputVariable	
FlowVariable	

Elements and Attributes	Description
Message	The target of the policy. Valid values are message, request, and response. When set to message, the policy conditionally retrieves the message object based on the attachment point of the policy. When attached to the request Flow, the policy resolves message to request, and when attached to the response Flow, the policy resolves message to response. Note that GenerateSAMLAssertion can only be attached to the TargetEndpoint request Flow. So, when using the Generate SAML Assertion policy, you should set this value to request.
XPath	An XPath expression that indicates the element on the outbound XML document to which the policy will attach the SAML assertion.
SignatureAlgorithm	SHA1 or SHA256
Subject	The unique identifier of the subject of the SAML assertion. If the optional ref attribute is present, then the value of Subject will be assigned at runtime based on the specified variable. If the optionalref attribute is not present, then the value of Subject will be used.
Template	If present, then the assertion will be generated by running this template, replacing everything denoted {} with the corresponding variable, and then digitally signing the result. The template is processed following the AssignMessage policy rules.

## SAML Assertion Validation

Policy Processing:

- The policy verifies that the inbound message request's media type is XML, by checking if the content type matches the formats `text/(.*)?xml` or `application/(.*)?xml`. If the media type is not XML, or if `IgnoreContentType` is not set, the policy raises a fault.
- The policy parses the XML. If parsing fails, it raises a fault.
- The policy validates the XML digital signature, using the values of `TrustStore` and `ValidateSigner` as described above. If validation fails, it raises a fault.
- The policy checks the current timestamp (if present) against the `NotBefore` and `NotOnOrAfter` elements in the assertion.

Successful completion of the policy ensures the following:

- The digital signature on the assertion is valid and was signed by a trusted CA.
- The assertion is valid for the current time period.
- The subject and issuer of the assertion would be extracted and set in flow variables. Other policies would use these values for additional authentication, such as checking if the subject name is valid, or passing it to a target system for validation.

Sample Schema for SAML Assertion Validation

### Sample Code

```
<!-- The policy will validate saml request, the saml assertion is extracted from variable defined in xpath -->
```

```

<ValidateSAMLAssertion async="false" continueOnError="false" enabled="true"
ignoreContentType = "false" xmlns="http://www.sap.com/apimgmt">
 <RemoveAssertion>>false</RemoveAssertion>
 <Source name="request">
 <Namespaces>
 <Namespace prefix="samlp">urn:oasis:names:tc:SAML:2.0:protocol</Namespace>
 <Namespace prefix="saml2">urn:oasis:names:tc:SAML:2.0:assertion</
Namespace>
 </Namespaces>
 <XPath>/samlp:Response/saml2:Assertion</XPath>
 </Source>
 <TrustStore>saml_trust_store</TrustStore>
</ValidateSAMLAssertion>

```

## Element Reference

Elements and Attributes	Description
name attribute	The name of the policy instance. The name must be unique in the organization. Characters you can use in the name are restricted to: A-Z0-9._\-\$%. However, the Management UI enforces additional restrictions, such as automatically removing characters that are not alphanumeric.
ignoreContentTypeattribute	A boolean that needs to be set to false. By default, the assertion will not be generated if the content type of the message is not an XML Content-Type. If this is set to true then the message will be treated as XML regardless of the Content-type.
Source	The target of the policy. Valid values are message, request, and response. When set to message, the policy conditionally retrieves the message object based on the attachment point of the policy. When attached to the request Flow, the policy resolves message to request, and when attached to the response Flow, the policy resolves message to response. Note that ValidateSAMLAssertion can only be attached to the ProxyEndpoint request Flow.
XPath	Child of Source. An XPath expression that indicates the element on the inbound XML document from which the policy can extract the SAML assertion.
Truststore	The name of the TrustStore that contains trusted X.509 certificates used to validate digital signatures on SAML assertions.
RemoveAssertion	A boolean that can be set to true or false. When true, the SAML assertion will be stripped from the request message before the message is forwarded to the backend service.

The following flow variables are available after the policy is executed:

### Flow Variables

Variable	Description
saml.id	The SAML assertion ID

Variable	Description
saml.issuer	The "Issuer" of the assertion, converted from its native XML type to a string
saml.subject	The "Subject" of the assertion, converted from its native XML type to a string
saml.valid	Returns true or false based on the result of the validity check
saml.attribute.{attribute_name}	The value of the named "Attribute" present in the assertion, converted from its native XML to a string
saml.attributeNames	The names of all the "Attributes" present in the assertion, in a comma-separated list
saml.issueInstant	IssueInstant
saml.subjectFormat	Subject format
saml.scmethod	Subject confirmation method
saml.scdaddress	Subject confirmation data address
saml.scdinresponse	Subject confirmation data in response
saml.scdrcpt	Subject confirmation data recipient
saml.authnSnooa	AuthnStatement SessionNotOnOrAfter
saml.authnContextClassRef	AuthnStatement AuthnContextClassRef
saml.authnInstant	AuthnStatement AuthInstant
saml.authnSessionIndex	AuthnStatement Session Index

### 1.5.1.4.1.26 Message Validation Policy

Validates a message and rejects it if it does not conform to the specified requirements.

This policy is used to:

- Validate any XML message against an XSD schema.
- Validate SOAP messages against a WSDL definition.
- Confirm JSON or XML is well-formed, based on content-type (if <ResourceURL> element is omitted)

You can attach the policy in the following locations

ProxyEndpoint				TargetEndpoint			
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	←Response
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows

### Sample Code

```
<!-- The policy will validate the response again the xsd, in the below policy
the soap message is validated to
contain a element with name "msg" and type string -->
<MessageValidation async="false" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <Element namespace="http://www.webserviceX.NET">string</Element>
 <Source>response</Source>
 <ResourceURL>xsd://validation.xsd</ResourceURL>
</MessageValidation>
Example validation.xsd
<?xml version="1.0" encoding="UTF-8"?><xs:schema xmlns:xs="http://
www.w3.org/2001/XMLSchema" attributeFormDefault="unqualified"
elementFormDefault="qualified" targetNamespace="http://www.webserviceX.NET">
 <xs:element name="msg" type="xs:string"/>
</xs:schema>
```

Elements and Attributes	Description
Name	The internal name of the policy. Characters you can use in the name are restricted to: A-Z0-9_!\-\$. Optionally, use the <DisplayName> element to label the policy in the UI proxy editor with a different, natural-language name.
continueOnError	Set to false to return an error when a policy fails. This is expected behavior for most policies. Set to true to have flow execution continue even after a policy fails.
enabled	Set to true to enforce the policy. Set to false to "turn off" the policy. The policy will not be enforced even if it remains attached to a flow.
async	This attribute is deprecated.
DisplayName	Use in addition to the name attribute to label the policy in the management UI proxy editor with a different, natural-language name. If you omit this element, then the value of the policy's name attribute is used.

Elements and Attributes	Description
Source	<p>Identifies the source message to be validated.</p> <p>If you do not provide a &lt;Source&gt; value, a value of message is used.</p> <p>If the &lt;Source&gt; variable cannot be resolved, or resolves to a non-message type, then one of the following occurs:</p> <p>If the source variable resolves to a null value in the message flow, a steps.messagevalidation.SourceMessageNotAvailable error code is thrown.</p> <p>If the source variable resolves to a non-message value, a steps.messagevalidation.NonMessageVariable error code is thrown.</p>
ResourceURL	<p>Identifies the XSD schema or WSDL definition to be used to validate the source message.</p> <p>If the WSDL does not have schemas or if the maximum import depth exceeds 10, message validation will fail.</p> <p>If a &lt;ResourceURL&gt; value is not specified, the message is checked for well-formed JSON or XML if the content-type is application/json or application/xml, respectively.</p> <p>Default: wsdl://&lt;name&gt;</p> <p>Presence: Optional</p> <p>Type: String</p>
SOAPMessage	<p>Provides the SOAP version against which to validate SOAP messages. &lt;SOAPMessage version="1.1/1.2"/&gt;</p>
Version	<p>Identifies the SOAP version against which to validate SOAP messages. Valid values: 1.1, 1.2, 1.1/1.2</p>
Element	<p>Specifies the root, or parent, element of the messages to be validated.</p> <p>&lt;Element namespace="http://finance.com/1999"&gt;PurchaseOrder&lt;/Element&gt;</p> <p>&lt;Element namespace="http://finance.com/2000"&gt;PurchaseOrder&lt;/Element&gt;</p>
namespace	<p>Provides the namespace of the root, or parent, element of the messages to be validated. Default: "http://sample.com"</p>

Message Validation policy type defines the following error codes:

Error code	Cause
InvalidResourceType	InvalidResourceType MessageValidation {0}: Invalid Resource Type {1}. It should be xsd or wsdl. Context {2}
ResourceCompileFailed	ResourceCompileFailed MessageValidation {0}: Failed to compile resource {1}. Context {2}
RootElementNameUnspecified	RootElementNameUnspecified MessageValidation {0}: RootElement name is not specified
InvalidRootElementName	InvalidRootElementName MessageValidation {0}: RootElement name {1} is invalid
NonMessageVariable	NonMessageVariable Variable {0} does not resolve to a Message
SourceMessageNotAvailable	SourceMessageNotAvailable {0} message is not available for MessageValidation: {1}
NoElements	Resource "{0}" has no element definitions
Failed	MessageValidation {0} failed with reason: "{1}"

### 1.5.1.4.1.27 Verify API Key

One of the mechanisms to prevent unauthorized access to APIs exposed over the internet is to use the verify API key policy.

The verify API key policy enforces verification of the application key in order to access your APIs.

API Management automatically generates API keys on behalf of applications. It enables API providers to view and approve API keys. By applying a policy of the type VerifyApiKey, you can enforce verification of API keys at runtime. This ensures that no application can access a protected API without a valid key.

The only setting required for the VerifyApiKey is the expected location of the API key. This policy can be attached to request or response stream of the proxy endpoint or target endpoint.

The schema for the Verify API Key policy is as follows:

#### Note

The below schema is not a working sample payload.

#### Code Syntax

```
<!-- The policy will prevent unauthorized users from call the api, only users
with valid app key will be able to access API. The policy expects the api key
to be sent as query param with name "key"-->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<VerifyApiKey async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <APIKey ref="request.queryparam.key"></APIKey>
</VerifyApiKey>
```

Elements & Attributes	Description
APIKey(Mandatory)	<p>The variable where the API key can be found.</p> <p>The API key is extracted from the request message by reference to a Flow variable. For example:</p> <pre>&lt;APIKey ref="request.queryparam.apikkey" /&gt;</pre> <p>If an application is expected to present the API key as the value of an HTTP header named APIKey, then set this value to request.header.APIKey.</p>

The policy populates several different types of flow variables, including:

- General
- App
- Developer
- Analytics

The following table lists the general flow variables populated by the Verify API Key policy.

These variables are all prefixed by: `verifyapikkey.{policy_name}`

For example: `verifyapikkey.{policy_name}.client_id`

General Flow Variables

Variable	Description
client_id	The consumer key (aka API key or app key) supplied by the requesting app
client_secret	The consumer secret associated with the consumer key
redirection_uris	Any redirect URIs in the request
developer.app.id	The ID of the developer app making the request
developer.app.name	The app name of the developer app making the request
developer.id	The ID of the developer registered as the owner of the requesting app
DisplayName	The value of the policy's <DisplayName> attribute
failed	Set to "true" when API Key validation fails.
apiproduct.name*	The name of the API product used to validate the request
apiproduct.developer.quota.limit*	The quota limit set on the API product, if any
apiproduct.developer.quota.interval*	The quota interval set on the API product, if any
apiproduct.developer.quota.timeunit*	The quota time unit set on the API product, if any



## Note

\* API product variables are populated automatically if the API products are configured with valid environment, proxies, and resources (derived from the proxy.pathsuffix).

The following flow variables containing information about the app are populated by the policy.

These variables are all prefixed by: `verifyapikey.{policy_name}.app.`

For example: `verifyapikey.{policy_name}.app.name`

### App Flow Variables

Variable	Description
name	The name of the app
id	The ID of the app
accessType	Unused by API Management
callbackUrl	The callback URL of the app, typically used only for OAuth
DisplayName	The app's display name
status	The app status, such as 'approved' or 'revoked'
apiproducts	An array containing the list of API products associated with the app
appType	The app type is "Developer"
created_at	The date/time stamp when the app was created
created_by	The e-mail address of the developer who created the app
last_modified_at	The date/time stamp when the app was last updated
last_modified_by	The e-mail address of the developer who last updated the app

The following flow variables containing information about the developer are populated by the policy.

These variables are all prefixed by: `verifyapikey.{policy_name}.developer.`

For example: `verifyapikey.{policy_name}.developer.id`

### Developer Flow Variables

Variable	Description
userName	The developer's user name
id	Returns <code>{org_name}@@@{developer_id}</code>

Variable	Description
firstName	The developer's first name
lastName	The developer's last name
e-mail	The developer's e-mail address
status	The developer's status, as active, inactive, or login_lock
apps	An array of apps associated with the developer
created_at	The date/time stamp when the developer was created
created_by	The e-mail address of the user who created the developer
last_modified_at	The date/time stamp when the developer was last modified
last_modified_by	The e-mail address of the user who modified the developer

The following variables are automatically populated in Analytics when a Verify API Key policy is enforced for a valid API key.

- apiproduct.name
- developer.app.name
- client\_id
- developer.id

During the policy execution, the following errors can occur:

Error Code

Error Name	HTTP Status	Cause
DeveloperStatusNotActive	401	The developer who created the Developer App that has the API key you are using has an inactive status. When an App Developer's status is set to inactive, any Developer Apps created by that developer are deactivated.
FailedToResolveAPIKey	401	The policy expects to find the API key in a variable that is specified in the policy's <APIKey> element. This error arises when the expected variable does not exist.
InvalidApiKey	401	An API key was received by API Management, but it is invalid. When API Management looks up the key in its database, it must exactly match the one that was sent in the request. If the API worked previously, make sure the key was not regenerated. If the key was regenerated, you will see this error if you try to use the old key.
InvalidApiKeyForGivenResource	401	An API key was received by API Management, and it is valid; however, it does not match an approved key in the Developer App associated with your API proxy through a Product.
invalid_client-app_not_approved	401	The Developer App associated with the API key is revoked.

Following errors can occur when you deploy a proxy containing this policy:

Deployment errors

Error name	Cause
SpecifyValueOrRefApiKey	The APIKey element does not have a value or key specified.

Following fault variables is set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	The fault variable [prefix] is oauthV2.  The [policy_name] is the name of the policy that threw the error.	oauthV2.VK-VerifyAPIKey.failed = true
fault.[error_name]	[error_name] = The specific error name to check for as listed in the table above.	fault.name Matches "FailedToResolveAPIKey"

Following is an example of an error response:

#### Sample Code

```
{
 "fault": {
 "faultstring": "Invalid ApiKey",
 "detail": {
 "errorcode": "oauth.v2.InvalidApiKey"
 }
 }
}
```

Following is an example of a fault rule:

#### Sample Code

```
<FaultRule name="FailedToResolveAPIKey">
 <Step>
 <Name>AM-FailedToResolveAPIKey</Name>
 </Step>
 <Condition>(fault.name Matches "FailedToResolveAPIKey") </Condition>
</FaultRule>
```

## 1.5.1.4.1.28 XML to JSON

API Management provides policies to convert messages from the extensible markup language (XML) format to JavaScript object notation (JSON) format. This policy is called the XML to JSON.

In an ideal scenario, you often pair a JSON to XML policy on the inbound request flow with an XML to JSON policy on the outbound response flow. By combining policies this way, a JSON API can be exposed for back end services that natively support only XML.

In cases where the APIs are consumed by diverse client apps that may require either JSON or XML, you can set the response format dynamically by configuring JSON to XML and XML to JSON policies to execute with conditions.

The policy can be attached in the following locations:

ProxyEndpoint				TargetEndpoint			←Response
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

### Code Syntax

```
<!-- The policy will convert xml response to json response and store in
variable named jsonresponse
, Attributes of a xml tag will be mapped with root "root_tag" and the
attributes inside root will have names with prefix as "attributes"-->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<XMLToJSON async="false" continueOnError="false" enabled="true" xmlns="http://
www.sap.com/apimgmt">
 <Options>
 <OutputSuffix>_SUFFIX</OutputSuffix>
 <OutputPrefix>PREFIX_</OutputPrefix>
 <AttributePrefix>BAR_</AttributePrefix>
 <AttributeBlockName>FOO_BLOCK</
AttributeBlockName>
 <TextNodeName>TEXT</TextNodeName>
 <TextAlwaysAsProperty>>true</
TextAlwaysAsProperty>
 <!-- The below three elements have to be used
in conjunction if there is a namespace in the xml that is to undergo
conversion -->
 <NamespaceSeparator/>
 <DefaultNamespaceNodeName/>
 <NamespaceBlockName/>
 <NullValue>NULL</NullValue>
 <RecognizeNull>>true</RecognizeNull>
 <RecognizeNumber>>true</RecognizeNumber>
 <RecognizeBoolean>>true</RecognizeBoolean>
 <TreatAsArray>
 <Path unwrap="false">custom/
 </TreatAsArray>
 </Options>
 <!-- The variable to which the converted JSON should be
assigned to -->
 <OutputVariable>response</OutputVariable>
 <!-- The variable that we want to convert to JSON -->
 <Source>response</Source>
</XMLToJSON>
```

Attribute Name	Description
Source (optional)	<p>The request or response variable that contains the XML message that you want to convert to JSON.</p> <p>Usually, you set this to request or response, depending on whether you need to convert an inbound XML request, or an outbound XML response, into JSON.</p> <p>If you don't define the Source, it is treated as message. If the source variable is unresolved, or resolves to a non-message type, the policy throws an error.</p> <p>Syntax: <code>&lt;Source&gt;request&lt;/Source&gt;</code></p>
OutputVariable (mandatory when the variable defined in the Source is a string)	<p>Stores the output of the XML to JSON format conversion. This is usually the same value as the source, that is, usually inbound XML request in converted to an inbound JSON request.</p> <p>If you do not specify an OutputVariable, the source is treated as OutputVariable. For example, if the source is response, then OutputVariable defaults to response.</p> <p>Syntax: <code>&lt;OutputVariable&gt;response&lt;/OutputVariable&gt;</code></p>
Options	<p>Use Options to have control over the conversion from XML to JSON.</p> <p>The following configuration elements can be added as children of the Options element. All options are optional, however, at a minimum, an empty Options element must be present for a policy to be valid.</p>

Attribute Name	Description
<Options>/<RecognizeBoolean>	<p data-bbox="584 353 1342 421">Allows the conversion to maintain boolean values instead of turning them into strings.</p> <p data-bbox="584 443 842 465">Valid values: true and false</p> <p data-bbox="584 488 820 510">The default value is true</p> <p data-bbox="584 539 935 562">If the policy configuration looks like:</p> <pre data-bbox="600 591 1398 645" style="background-color: #f0f0f0; padding: 5px;"> &lt;RecognizeBoolean&gt;true&lt;/RecognizeBoolean&gt; </pre> <p data-bbox="584 667 847 689">Consider an XML example:</p> <pre data-bbox="600 712 1398 842" style="background-color: #f0f0f0; padding: 5px;"> &lt;x&gt;   &lt;y&gt;false&lt;/y&gt;   &lt;z&gt;value&lt;/z&gt; &lt;/x&gt; </pre> <p data-bbox="584 864 703 887">If true, then:</p> <pre data-bbox="600 909 1398 1093" style="background-color: #f0f0f0; padding: 5px;"> {   "x": {     "y": false,     "z": "value"   } } </pre> <p data-bbox="584 1115 711 1137">If false, then:</p> <pre data-bbox="600 1160 1398 1344" style="background-color: #f0f0f0; padding: 5px;"> {   "x": {     "y": "false",     "z": "value"   } } </pre>

Attribute Name	Description
<Options>/<RecognizeNumber>	<p data-bbox="584 353 1358 416">Allows the number fields in the XML payload to retain their original format if the value is true.</p> <p data-bbox="584 443 828 465">Valid values: true or false</p> <p data-bbox="584 492 764 515">Default value: true</p> <p data-bbox="584 542 935 564">If the policy configuration looks like:</p> <pre data-bbox="600 591 1396 645" style="background-color: #f0f0f0; padding: 5px;"> &lt;RecognizeNumber&gt;true&lt;/RecognizeNumber&gt; </pre> <p data-bbox="584 667 847 689">Consider an XML example:</p> <pre data-bbox="600 712 1396 842" style="background-color: #f0f0f0; padding: 5px;"> &lt;x&gt;   &lt;y&gt;999&lt;/y&gt;   &lt;z&gt;value&lt;/z&gt; &lt;/x&gt; </pre> <p data-bbox="584 864 703 887">If true, then:</p> <pre data-bbox="600 909 1396 1093" style="background-color: #f0f0f0; padding: 5px;"> {   "x": {     "y": 999     "z": "value"   } } </pre> <p data-bbox="584 1115 703 1137">If false, then:</p> <pre data-bbox="600 1160 1396 1344" style="background-color: #f0f0f0; padding: 5px;"> {   "x": {     "y": "999"     "z": "value"   } } </pre>

Attribute Name	Description
<Options>/<RecognizeNull>	<p>Allows you to convert empty values to null values.</p> <p>Valid values: true and false</p> <p>The default value is false</p> <p>If the policy configuration looks like:</p>
	<pre data-bbox="600 568 1107 591">&lt;RecognizeNull&gt;true&lt;/RecognizeNull&gt;</pre>
	<p>Consider an XML example:</p>
	<pre data-bbox="600 689 804 790">&lt;x&gt;   &lt;y&gt;&lt;/y&gt;   &lt;z&gt;value&lt;/z&gt; &lt;/x&gt;</pre>
	<p>If true, then:</p>
	<pre data-bbox="600 889 890 1039">{   "x": {     "y": null     "z": "value"   } }</pre>
	<p>If false, then:</p>
	<pre data-bbox="600 1137 890 1288">{   "x": {     "y": {},     "z": "value"   } }</pre>
<Options>/<NullValue>	<p>Indicates a null value. By default the value is NULL.</p> <p>Syntax: &lt;NullValue&gt;NULL&lt;/NullValue&gt;</p>
<Options>/<NamespaceSeparator> <Options>/<NamespaceBlock- Name>	<p>Use the three elements NamespaceSeparator, NamespaceBlockName, and DefaultNamespaceNodeName together.</p> <p>If the policy configuration looks like:</p>
	<pre data-bbox="600 1592 1326 1715">&lt;NamespaceBlockName&gt;#namespaceblock&lt;/ NamespaceBlockName&gt;   &lt;DefaultNamespaceNodeName&gt;\$defaultname&lt;/ DefaultNamespaceNodeName&gt;   &lt;NamespaceSeparator&gt;:&lt;/NamespaceSeparator&gt;</pre>
	<p>Consider the following XML example:</p>
	<pre data-bbox="600 1814 1251 1915">&lt;x xmlns="http://abc.com" xmlabc:ns1="http:// abc1.com"&gt;   &lt;abc1:y&gt;value&lt;/abc1:y&gt; &lt;/x&gt;</pre>



Attribute Name	Description
<Options>/<DefaultNamespaceNodeName>	<p data-bbox="584 344 1396 383">If NamespaceSeparator isn't specified, the following JSON structure is generated:</p> <pre data-bbox="600 405 1380 555"> {   "x": {     "y": "value"   } } </pre> <p data-bbox="584 577 1396 678">If the elements NamespaceSeparator, NamespaceBlockName, and DefaultNamespaceName are specified as <b>: #namespaceblock</b>, and <b>\$defaultname</b> respectively, then the following JSON structure is generated:</p> <pre data-bbox="600 701 1380 958"> {   "x": {     "&amp;namespaces": {       "%": "http://abc.com",       "abc1": "http://abc1.com"     },     "abc1:y": "value"   } } </pre>
<Options>/<OutputPrefix>	<p data-bbox="584 981 1396 1019">Use the two elements OutputPrefix and OutputSuffix together.</p> <p data-bbox="584 1041 662 1070">Syntax:</p> <pre data-bbox="600 1093 1380 1171"> &lt;OutputSuffix&gt;_SUFFIX&lt;/OutputSuffix&gt; &lt;OutputPrefix&gt;PREFIX_&lt;/OutputPrefix&gt; </pre> <p data-bbox="584 1193 1125 1223">Consider the following XML example: <code>&lt;x&gt;value&lt;/x&gt;</code></p> <p data-bbox="584 1245 1396 1305">If both the attributes are specified as defined in the XML to JSON example, the following JSON structure is generated:</p> <pre data-bbox="600 1328 1380 1429"> PREFIX_{   "x": "value" }_SUFFIX </pre> <p data-bbox="584 1451 1316 1480">If only OutputPrefix is specified, the following JSON structure is generated:</p> <pre data-bbox="600 1503 1380 1603"> PREFIX_{   "x" : "value" } </pre> <p data-bbox="584 1626 1316 1655">If only OutputSuffix is specified, the following JSON structure is generated:</p> <pre data-bbox="600 1677 1380 1778"> {   "x" : "value" }_SUFFIX </pre> <p data-bbox="584 1800 1396 1861">If neither OutputPrefix nor OutputSuffix is specified, the following JSON structure is generated:</p> <pre data-bbox="600 1883 1380 1951"> {   "x": "value" } </pre>

Attribute Name	Description
<Options>/<OutputSuffix>	}
<Options>/<AttributeBlockName>	Use the two attributes AttributeBlockName and AttributePrefix together, to group the values into a JSON block and append prefixes to the attribute names.
<Options>/<AttributePrefix>	<p data-bbox="584 517 1182 580">Consider the following XML example: &lt;a att1="value1" att2="value2" /&gt;</p> <p data-bbox="584 607 935 629">If the policy configuration looks like:</p> <pre data-bbox="600 663 1326 719" style="background-color: #f0f0f0; padding: 5px;"> &lt;AttributeBlockName&gt;FOO_BLOCK&lt;/AttributeBlockName&gt; &lt;AttributePrefix&gt;BAR_&lt;/AttributePrefix&gt;</pre>
	Then, the following JSON structure is generated:
	<pre data-bbox="600 808 719 1021" style="background-color: #f0f0f0; padding: 5px;"> {   "a": {     "FOO_BLOCK": {       "BAR_att1": "value1",       "BAR_att2": "value2"     }   } }</pre>
	If only AttributeBlockName is specified, the following JSON structure is generated:
	<pre data-bbox="600 1111 1015 1323" style="background-color: #f0f0f0; padding: 5px;"> {   "a": {     "FOO_BLOCK": {       "att1": "value1",       "att2": "value2"     }   } }</pre>
	If only AttributePrefix is specified, the following JSON structure is generated:
	<pre data-bbox="600 1413 1078 1603" style="background-color: #f0f0f0; padding: 5px;"> {   "a": {     "BAR_att1": "value1",     "BAR_att2": "value2"   } }</pre>
	If neither value is specified, the following JSON structure is generated:
	<pre data-bbox="600 1693 959 1816" style="background-color: #f0f0f0; padding: 5px;"> "a": {   "att1": "value1",   "att2": "value2" }</pre>
<Options>/<TextAlwaysAsProperty>	<p data-bbox="584 1861 1318 1883">Use the two elements TextAlwaysAsProperty and TextNodeName together.</p> <p data-bbox="584 1917 807 1939">Valid values: true/false</p>

Attribute Name	Description
----------------	-------------

<Options>/<TextNodeName>

Default is false.

If set to true, the content of the XML element is converted to a string property.

If the policy configuration looks like:

```
<TextNodeName>TEXT</TextNodeName>
 <TextAlwaysAsProperty>true</
TextAlwaysAsProperty>
```

For the following XML:

```
<a>
 value1
 <c>value2<d>value3</d>value4</c>

```

If TextAlwaysAsProperty is set to true and TextNodeName specified as TEXT, the following JSON structure is generated:

```
{
 "a": {
 "b": {
 "TEXT": "value1"
 },
 "c": {
 "TEXT": [
 "value2",
 "value4"
],
 "d": {
 "TEXT": "value3"
 }
 }
 }
}
```

If TextAlwaysAsProperty is set to false and TextNodeName specified as TEXT, the following JSON structure is generated:

```
{
 "a": {
 "b": "value1",
 "c": {
 "TEXT": [
 "value2",
 "value4"
],
 "d": "value3"
 }
 }
}
```

**Attribute Name****Description**`<Options>/<TreatAsArray>`

Enables you to streamline the values from an XML document into a JSON array. This attribute is useful when the number of child elements vary from one to many, and you want to ensure the values are always in an array.

Syntax:

```
<Options>
 <TreatAsArray>
 <Path unwrap="true">employers/employername/
employees/name</Path>
 </TreatAsArray>
</Options>
```

Consider the following xml example:

**Sample Code**

```
#Example 1
<employers>
 <employername>
 <name>employer1</name>
 <employees>
 <name>emp1</name>
 <name>emp2</name>
 </employees>
 </employername>
</employers>
Output
{
 "employers" : {
 "employername" : {
 "name" : "employer1",
 "employees" : {
 "name" : [
 "emp1",
 "emp2"
]}...
 }
}

Example 2
<employers>
 <employername>
 <name>employer1</name>
 <employees>
 <name>emp1</name>
 </employees>
 </employername>
</employers>
Output
{
 "employers" : {
 "employername" : {
 "name" : "employer1",
 "employees" : {
 "name" : "emp1"
 }
 }
 }
}
```

## Attribute Name

## Description

By default, XML to JSON policy puts multiple child values into an array (example 1). However, when there is only one child, the policy places it in a string. In such cases `<TreatAsArray>/<Path>` element allows you to control the output.

Using the `<TreatAsArray>/<Path>` element you can ensure that values for `<name>` is always put oin an array, even for a single value. You configure this by identifying the Path to the element whose values you want to put in an array. like: (consider example 2)

```
<TreatAsArray>
 <Path>employers/employername/employees/name</
Path>
</TreatAsArray>
```

### Sample Code

```
<employers>
 <employername>
 <name>employer1</name>
 <employees>
 <name>empl</name>
 </employees>
 </employername>
</employers>
Output
{
 "employers" : {
 "employername" : {
 "name" : "employer1",
 "employees" : {
 "name" : [
 "empl",
]}...
 }
 }
}
```

Also, in the above output, `<employername>` element and the `<name>` element for employees are unnecessary. We can unwrap them using the following syntax:

### Code Syntax

```
<TreatAsArray>
 <Path unwrap="true">employers/
employername/employees/name</Path>
 <Path unwrap="true">employers/
employername/employees/name</Path>
</TreatAsArray>
```

Output:

### Sample Code

```
{
 "employers" : [{
 "name" : "employer1",
```

Attribute Name	Description
	<pre>"employees" : [ "emp1", "emp2" ] }]...</pre>

During the policy execution, the following errors can occur:

Error Code

Error Name	HTTP Status	Cause
EitherOptionOrFormat	500	See the fault string.
UnknownFormat	500	See the fault string.
FormatUnavailable	500	See the fault string.
SourceUnavailable	500	The variable specified in the <Source> element has to exist.
ExecutionFailed	500	See the fault string. Be sure the incoming message contains valid XML.
InvalidSourceType	500	See the fault string.
InCompatibleTypes	500	See the fault string.
OutputVariablesNotAvailable	500	See the fault string.

Following fault variables is set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	The [prefix] is xmltojson..  The [policy_name] is the name of the policy that threw the error.	xmltojson.XMLtoJSON-1.failed = true
fault.[error_name]	[error_name] = The specific error name to check for as listed in the table above.	fault.name Matches "SourceUnavailable"

Following is an example of an error response:

```

❏ Sample Code

{
 "fault": {
 "faultstring": "XMLToJSON[XMLtoJSON_1]: Source xyz is not available",
 "detail": {
 "errorcode": "steps.xml2json.SourceUnavailable"
 }
 }
}

```

Following is an example of a fault rule:

## Sample Code

```
<faultrule name="VariableOfNonMsgType"></faultrule><FaultRule name="XML to
JSON Faults">
 <Step>
 <Name>AM-SourceUnavailableMessage</Name>
 <Condition>(fault.name Matches "SourceUnavailable") </Condition>
 </Step>
 <Step>
 <Name>AM-BadXML</Name>
 <Condition>(fault.name = "ExecutionFailed")</Condition>
 </Step>
 <Condition>(xmltojson.XMLtoJSON_1.failed = true) </Condition>
</FaultRule>
```

## Related Information

[JSON to XML \[page 290\]](#)

### 1.5.1.4.1.29 XSL Transform

Extensible stylesheet language transformations (XSLT) is a language that is used to convert documents from one XML format to another. This policy is used in applications that support XML but require a different XML-format for the same data.

The XSL Transformation policy is executed as a processing step in an API proxy flow. The XSLT is implemented via an xsl file that is stored in the API proxy bundle under `APIProxy/FileResources/<policyname>.xsl`. The XSL policy references this XSL file.

The XSL policy requires two inputs:

- The name of an XSLT stylesheet (which contains a set of transformation rules) stored in the API proxy bundle under `APIProxy/FileResources`
- The source of the XML to be transformed (typically a request or response message)

#### Note

`<xsl:include>` and `<xsl:import>` are not supported in the xslt code used in this policy.

You can attach this policy in the following locations:

ProxyEndpoint				TargetEndpoint			
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	←Response
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

### Code Syntax

```
<!-- The policy will take the read the xml response and transform the xsl and
assign the transformed xml to the output variable outVar
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<XSL async="true" continueOnError="false" enabled="true" xmlns="http://
www.sap.com/apimgmt">
 <OutputVariable>outVar</OutputVariable>
 <ResourceURL>xsl://XSLTransform.xsl</ResourceURL>
 <Source>response</Source>
</XSL>
example : XSLTransform.xsl
<?xml version="1.0" encoding="UTF-8"?><xsl:stylesheet xmlns:xsl="http://
www.w3.org/1999/XSL/Transform" version="1.0">
 <xsl:template match="/hello-world">
 <HTML>
 <HEAD>
 <TITLE/>
 </HEAD>
 <BODY>
 <H1>
 <xsl:value-of select="greeting"/>
 </H1>
 <xsl:apply-templates select="greeter"/>
 </BODY>
 </HTML>
 </xsl:template>
 <xsl:template match="greeter">
 <DIV>from <I>
 <xsl:value-of select="."/>
 </I>
 </DIV>
 </xsl:template>
</xsl:stylesheet>
```

#### Elements and Attributes

#### Description

Source (Optional)

Contains the message from which information needs to be extracted. Usually this value is set to request or response, depending on whether the message to be transformed is inbound or outbound.

- If source is missing, it is treated as a simple message. For example, `<Source>message</Source>`.
- If the source variable cannot be resolved, or resolves to a non-message type, the transformation step fails.

OutputVariable (Optional)

A variable that stores the output of the transformation. The OutputVariable cannot be of Message type, that is, it cannot be 'message', 'request', or 'response'. You should set this element to be a custom variable, and then consume that variable.

To replace the message content with the output of the transformation, delete this element.



Elements and Attributes		Description	
ResourceURL (Mandatory)		The XSLT file to be used for transforming the message	
Parameters(Optional)	ignoreUnresolvedVariables (Optional)	<p>Ignores any unresolved variable errors in the XSLT script instructions.</p> <p>Valid values: true/false</p> <p>Default value: false</p>	
	Parameter (Optional)	name (Mandatory)	Name of a custom parameter. Note that with name you can only use one of the optional parameters listed below.
		ref(Optional)	Specifies the reference that sources the value from a variable.
		value (Optional)	Value of the parameter

During the policy execution, the following errors can occur:

Error Cause

Error Name	Cause
XSLSourceMessageNotAvailable	{0} message is not available for XSL: {1}
XSLEvaluationFailed	Evaluation of XSL {0} failed with reason: "{1}"
XSLVariableResolutionFailed	Failed to resolve variable {0}
XSLInvalidResourceType	XSL {0}: Resource type must be xsl. Context {1}
XSLEmptyResourceUrl	Resource Url cannot be empty in XSL {0}

### 1.5.1.4.1.30 XML Threat Protection

API Management enables developers to address XML vulnerabilities and minimize attacks on API. Further, it allows you to detect XML payload attacks based on configured limits and screen against XML threats by using the following approaches:

- Validating messages against the XML schema (.xsd)
- Evaluating message content for specific blocklisted keywords or patterns
- Detecting corrupt or malformed messages before such messages are parsed

You can attach this policy in the following locations:

ProxyEndpoint			TargetEndpoint				
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow	PostFlow	← Response
	•	•	•	•	•	•	
	•	•	•	•	•	•	
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

### Code Syntax

```
<XMLThreatProtection async="false" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <NameLimits>
 <Element>20</Element>
 <Attribute>20</Attribute>
 <NamespacePrefix>20</NamespacePrefix>
 <ProcessingInstructionTarget>20</ProcessingInstructionTarget>
 </NameLimits>
 <Source>request</Source>
 <StructureLimits>
 <NodeDepth>6</NodeDepth>
 <AttributeCountPerElement>3</AttributeCountPerElement>
 <NamespaceCountPerElement>5</NamespaceCountPerElement>
 <ChildCount includeComment="true" includeElement="true"
includeProcessingInstruction="true" includeText="true">5</ChildCount>
 </StructureLimits>
 <ValueLimits>
 <Text>10</Text>
 <Attribute>12</Attribute>
 <NamespaceURI>12</NamespaceURI>
 <Comment>12</Comment>
 <ProcessingInstructionData>12</ProcessingInstructionData>
 </ValueLimits>
</XMLThreatProtection>
```

Attribute Name	Description
Source	<p>Indicates the message that needs to be screened for XML payload attacks. If it is set to request, you must validate the inbound requests from client apps. If it is set to message, the element automatically evaluates the request or response message when the message is attached to a request flow or a response flow respectively.</p> <p>&lt;Source&gt;response&lt;/Source&gt;</p>

Attribute Name	Description
Name Limits (Optional)	<p data-bbox="544 344 628 367">Element</p> <p data-bbox="703 344 1398 405">NameLimits indicates the character limits that need to be checked and enforced by the policy.</p> <p data-bbox="703 427 1398 488">The NameLimits &lt;Element&gt; element indicates the limit on the maximum number of characters allowed in an element name.</p> <p data-bbox="703 510 1398 571">If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.</p> <div data-bbox="703 593 1398 1234" style="background-color: #f0f0f0; padding: 10px;"> <p data-bbox="730 602 946 636">↔ Sample Code</p> <p data-bbox="730 651 820 674">Example</p> <p data-bbox="730 696 1038 719">For the following example XML:</p> <pre data-bbox="746 757 1257 887" style="background-color: #e0e0e0; padding: 5px;"> &lt;book category="WEB"&gt;   &lt;title&gt;XML for Beginners&lt;/title&gt;   &lt;author&gt;Adam J. Smith&lt;/author&gt;   &lt;year&gt;2010&lt;/year&gt; &lt;/book&gt; </pre> <p data-bbox="730 920 1398 981">The policy snippet below validates that the element names do not exceed the specified character limit.</p> <pre data-bbox="746 1016 1331 1193" style="background-color: #e0e0e0; padding: 5px;"> &lt;NameLimits&gt;   &lt;Element&gt;15&lt;/Element&gt;   &lt;Attribute&gt;15&lt;/Attribute&gt;   &lt;NamespacePrefix&gt;15&lt;/NamespacePrefix&gt;   &lt;ProcessingInstructionTarget&gt;15&lt;/ProcessingInstructionTarget&gt; &lt;/NameLimits&gt; </pre> </div>

## Attribute Name

## Description

Attribute

Indicates a limit on the maximum number of characters allowed in an attribute name within the XML document.

If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.

### ↔ Sample Code

Example

For the following example XML:

```
<book category="WEB">
 <title>XML for Beginners</title>
 <author>Adam J. Smith</author>
 <year>2010</year>
</book>
```

The policy snippet below validates that the attribute name does not exceed the specified character limit.

```
<NameLimits>
 <Element>15</Element>
 <Attribute>15</Attribute>
 <NamespacePrefix>15</NamespacePrefix>
 <ProcessingInstructionTarget>15</
ProcessingInstructionTarget>
</NameLimits>
```

Namespace-  
Prefix

Indicates a limit on the maximum number of characters allowed in the namespace prefix within the XML document.

If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.

### ↔ Sample Code

Example

For the following example XML: `<ns1:myelem xmlns:abc="http://abc.com" />`

The policy snippet below validates that the namespace prefix **abc** does not exceed the specified character limit.

```
<NameLimits>
 <Element>15</Element>
 <Attribute>15</Attribute>
 <NamespacePrefix>15</NamespacePrefix>
 <ProcessingInstructionTarget>15</
ProcessingInstructionTarget>
</NameLimits>
```

Attribute Name	Description
ProcessingInstructionTarget	<p>Indicates a limit on the maximum number of characters allowed in the target of any processing instructions within the XML document.</p> <p>If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.</p>
<div style="background-color: #f0f0f0; padding: 10px;"> <p><b>Sample Code</b></p> <p>Example</p> <p>For the following example XML: <code>&lt;?xml-doc type="text/xsl" href="doc.xsl" ?&gt;</code></p> <p>The policy snippet below validates that the processing instruction target <code>xml-doc</code> does not exceed the specified character limit.</p> <pre data-bbox="746 801 1331 981">&lt;NameLimits&gt;   &lt;Element&gt;15&lt;/Element&gt;   &lt;Attribute&gt;15&lt;/Attribute&gt;   &lt;NamespacePrefix&gt;15&lt;/NamespacePrefix&gt;   &lt;ProcessingInstructionTarget&gt;15&lt;/ProcessingInstructionTarget&gt; &lt;/NameLimits&gt;</pre> </div>	
StructuralLimits (Optional)	<p><b>NodeDepth</b></p> <p>StructuralLimits indicates the structural limits that need to be checked and enforced by the policy.</p> <p>The StructuralLimits <code>&lt;NodeDepth&gt;</code> element indicates the maximum node depth that is allowed within an XML document.</p> <p>If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.</p>
<div style="background-color: #f0f0f0; padding: 10px;"> <p><b>Sample Code</b></p> <p>Example</p> <pre data-bbox="746 1413 1270 1686">&lt;StructureLimits&gt;   &lt;NodeDepth&gt;6&lt;/NodeDepth&gt;   &lt;AttributeCountPerElement&gt;3&lt;/AttributeCountPerElement&gt;   &lt;NamespaceCountPerElement&gt;5&lt;/NamespaceCountPerElement&gt;   &lt;ChildCount includeComment="true" includeElement="true" includeProcessingInstruction="true" includeText="true"&gt;5&lt;/ChildCount&gt; &lt;/StructureLimits&gt;</pre> </div>	

Attribute Name	Description
Attribute-CountPerElement	<p>Indicates the maximum number of attributes allowed for any element within an XML document.</p> <p>If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.</p>
<div data-bbox="730 517 944 551"> <p>↔ Sample Code</p> </div> <div data-bbox="730 566 820 593"> <p>Example</p> </div> <div data-bbox="730 616 1037 642"> <p>For the following example XML:</p> </div> <div data-bbox="746 678 1257 804" style="background-color: #f0f0f0; padding: 10px;"> <pre>&lt;book category="WEB"&gt;   &lt;title&gt;XML for Beginners&lt;/title&gt;   &lt;author&gt;Adam J. Smith&lt;/author&gt;   &lt;year&gt;2010&lt;/year&gt; &lt;/book&gt;</pre> </div> <div data-bbox="730 837 1353 936"> <p>The policy snippet below validates that the elements book, title, author, and year do not have more than <b>3</b> attributes each. The attributes used for defining namespaces are not counted.</p> </div> <div data-bbox="746 972 1270 1249" style="background-color: #f0f0f0; padding: 10px;"> <pre>&lt;StructureLimits&gt;   &lt;NodeDepth&gt;6&lt;/NodeDepth&gt;   &lt;AttributeCountPerElement&gt;3&lt;/ AttributeCountPerElement&gt;   &lt;NamespaceCountPerElement&gt;5&lt;/ NamespaceCountPerElement&gt;   &lt;ChildCount includeComment="true" includeElement="true" includeProcessingInstruction="true" includeText="true"&gt;5&lt;/ChildCount&gt; &lt;/StructureLimits&gt;</pre> </div>	

Attribute Name	Description
Namespace-CountPerElement	<p>Indicates the maximum number of namespace definitions allowed for any element within an XML document.</p> <p>If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.</p>

**Sample Code**

Example

For the following example XML:

```

<a1 attr1="value1" attr2="value2">
 <a2 xmlns="http://sap.com"
 xmlns:abc="http://abc.com" one="1"
 abc:two="2" />
</a1>

```

The policy snippet below validates that for the elements **a1** and **a2**, the number of namespace definitions are limited to **5** each. In the above example, the **a1** element has **0** namespace definitions and the **a2** element has **2** namespace definitions: **xmlns="http://sap.com"** and **xmlns:abc="http://abc.com"**.

```

<StructureLimits>
 <NodeDepth>6</NodeDepth>
 <AttributeCountPerElement>3</
 AttributeCountPerElement>
 <NamespaceCountPerElement>5</
 NamespaceCountPerElement>
 <ChildCount includeComment="true"
 includeElement="true"
 includeProcessingInstruction="true"
 includeText="true">5</ChildCount>
</StructureLimits>

```

**Attribute Name****Description**

ChildCount

Specifies the maximum number of child elements allowed for any element within an XML document.

If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.

The ChildCount element contains the following attributes:

- includeComment (Default: true)
- includeElement (Default: true)
- includeProcessingInstructions (Default: true)
- includeText (Default: true)

**Sample Code**

Example

```
<StructureLimits>
 <NodeDepth>6</NodeDepth>
 <AttributeCountPerElement>3</
AttributeCountPerElement>
 <NamespaceCountPerElement>5</
NamespaceCountPerElement>
 <ChildCount includeComment="true"
includeElement="true"
includeProcessingInstruction="true"
includeText="true">5</ChildCount>
</StructureLimits>
```



Attribute Name		Description
ValueLimits (Optional)	Text	<p data-bbox="707 344 1394 405">ValueLimits indicates the character limits for values to be checked and enforced by the policy.</p> <p data-bbox="707 427 1394 488">The ValueLimits &lt;Text&gt; element indicates the character limit for any text nodes within an XML document.</p> <p data-bbox="707 510 1394 568">If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.</p> <div data-bbox="707 591 1394 1301" style="background-color: #f0f0f0; padding: 10px;"> <p data-bbox="730 600 946 633">↔ Sample Code</p> <p data-bbox="730 651 820 674">Example</p> <p data-bbox="730 696 1038 719">For the following example XML:</p> <pre data-bbox="746 757 1257 891" style="background-color: #e0e0e0; padding: 5px;"> &lt;book category="WEB"&gt;   &lt;title&gt;XML for Beginners&lt;/title&gt;   &lt;author&gt;Adam J. Smith&lt;/author&gt;   &lt;year&gt;2010&lt;/year&gt; &lt;/book&gt; </pre> <p data-bbox="730 920 1362 1021">The policy snippet below validates that the element text values <b>XML for Beginners</b>, <b>Adam J. Smith</b>, and <b>2010</b> do not exceed <b>20</b> characters each.</p> <pre data-bbox="746 1055 1241 1256" style="background-color: #e0e0e0; padding: 5px;"> &lt;ValueLimits&gt;   &lt;Text&gt;20&lt;/Text&gt;   &lt;Attribute&gt;10&lt;/Attribute&gt;   &lt;NamespaceURI&gt;15&lt;/NamespaceURI&gt;   &lt;Comment&gt;10&lt;/Comment&gt;   &lt;ProcessingInstructionData&gt;10&lt;/ProcessingInstructionData&gt; &lt;/ValueLimits&gt; </pre> </div>

Attribute Name	Description
Attribute	<p>Indicates the character limit for any attribute values within an XML document.</p> <p>If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.</p> <div data-bbox="703 506 1396 1178" style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p><b>↔ Sample Code</b></p> <p>Example</p> <p>For the following example XML:</p> <pre data-bbox="746 678 1257 801">&lt;book category="WEB"&gt;   &lt;title&gt;XML for Beginners&lt;/title&gt;   &lt;author&gt;Adam J. Smith&lt;/author&gt;   &lt;year&gt;2010&lt;/year&gt; &lt;/book&gt;</pre> <p>The policy snippet below validates that the attribute value <b>WEB</b> does not exceed <b>10</b> characters.</p> <pre data-bbox="746 943 1241 1137">&lt;ValueLimits&gt;   &lt;Text&gt;20&lt;/Text&gt;   &lt;Attribute&gt;10&lt;/Attribute&gt;   &lt;NamespaceURI&gt;15&lt;/NamespaceURI&gt;   &lt;Comment&gt;10&lt;/Comment&gt;   &lt;ProcessingInstructionData&gt;10&lt;/ProcessingInstructionData&gt; &lt;/ValueLimits&gt;</pre> </div>
NamespaceURI	<p>Indicates the character limit for any namespace URIs within an XML document.</p> <p>If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.</p> <div data-bbox="703 1366 1396 1904" style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p><b>↔ Sample Code</b></p> <p>Example</p> <p>For the following example XML: <code>&lt;ns:my_element xmlns:ns="http://ns.com" /&gt;</code></p> <p>The policy snippet below validates that the namespace URI value <b>http://ns.com</b> does not exceed <b>15</b> characters.</p> <pre data-bbox="746 1664 1241 1859">&lt;ValueLimits&gt;   &lt;Text&gt;20&lt;/Text&gt;   &lt;Attribute&gt;10&lt;/Attribute&gt;   &lt;NamespaceURI&gt;15&lt;/NamespaceURI&gt;   &lt;Comment&gt;10&lt;/Comment&gt;   &lt;ProcessingInstructionData&gt;10&lt;/ProcessingInstructionData&gt; &lt;/ValueLimits&gt;</pre> </div>

Attribute Name	Description
Comment	<p>Indicates the character limit for any comment within an XML document.</p> <p>If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.</p> <div data-bbox="705 506 1396 1238"><h3 data-bbox="730 517 943 548">Sample Code</h3><p data-bbox="730 566 818 586">Example</p><p data-bbox="730 616 1038 636">For the following example XML:</p><pre data-bbox="746 678 1257 826">&lt;book category="WEB"&gt;   &lt;!-- This is a comment --&gt;   &lt;title&gt;XML for Beginners&lt;/title&gt;   &lt;author&gt;Adam J. Smith&lt;/author&gt;   &lt;year&gt;2010&lt;/year&gt; &lt;/book&gt;</pre><p data-bbox="730 864 1366 960">The value of the &lt;comment&gt; element in the policy snippet below validates that the comment text <b>This is a comment</b> does not exceed <b>10</b> characters.</p><pre data-bbox="746 1003 1241 1200">&lt;ValueLimits&gt;   &lt;Text&gt;20&lt;/Text&gt;   &lt;Attribute&gt;10&lt;/Attribute&gt;   &lt;NamespaceURI&gt;15&lt;/NamespaceURI&gt;   &lt;Comment&gt;10&lt;/Comment&gt;   &lt;ProcessingInstructionData&gt;10&lt;/ProcessingInstructionData&gt; &lt;/ValueLimits&gt;</pre></div>

Attribute Name	Description
ProcessingInstructionData	<p>Indicates the character limit for any processing instruction text within an XML document.</p> <p>If you do not specify a limit, the policy applies a default value of -1, which denotes no limit.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p><b>Sample Code</b></p> <p>Example</p> <p>For the following example XML: <code>&lt;?xml-doc type="text/xsl" href="doc.xsl" ?&gt;</code></p> <p>The policy snippet below validates that the processing instruction text <code>type="text/xsl" href="doc.xsl"</code> does not exceed <b>10</b> characters.</p> <pre>&lt;ValueLimits&gt;   &lt;Text&gt;20&lt;/Text&gt;   &lt;Attribute&gt;10&lt;/Attribute&gt;   &lt;NamespaceURI&gt;15&lt;/NamespaceURI&gt;   &lt;Comment&gt;10&lt;/Comment&gt;   &lt;ProcessingInstructionData&gt;10&lt;/ProcessingInstructionData&gt; &lt;/ValueLimits&gt;</pre> </div>

XML Threat Protection policy type defines the following error codes:

Error Code	Cause
ExecutionFailed	<p>Errors that occur when specific thresholds set in the policies are exceeded.</p> <p>These errors include <b>node depth, attribute count, child count, namespace count, element name length, attribute name and value length, namespace prefix and URL length, processing instruction name and data length, comment length, and text length.</b></p>
NodeDepthExceeded	This error occurs when the maximum depth of XML elements allowed in an XML payload is exceeded.
AttrCountExceeded	This error occurs when the maximum number of attributes allowed in a single element is exceeded.
ChildCountExceeded	This error occurs when the maximum number of child elements allowed in an XML payload is exceeded.
NSCountExceeded	This error occurs when the number of name spaces allowed in a single element is exceeded.
ElemNameExceeded	This error occurs when the maximum string length allowed in an XML tag is exceeded.

Error Code	Cause
AttrNameExceeded	This error occurs when the maximum length allowed for an attribute name is exceeded.
AttrValueExceeded	This error occurs when the maximum length allowed for an attribute value is exceeded.
NSPrefixExceeded	This error occurs when the namespace prefix length is exceeded.
NSURIXceeded	This error occurs when the namespace URL length is exceeded.
PITargetExceeded	This error occurs when the process instruction name length is exceeded.
PIDataExceeded	The allowed processing instruction data length is exceeded.
CommentExceeded	This error occurs when the maximum length allowed for a comment is exceeded.
TextExceeded	This error occurs when the maximum length allowed for text is exceeded.
SourceUnavailable	This error occurs when the message variable mentioned in the source element is either unavailable in the specific flow where the policy is being executed or it does not have a valid value (request, response, or message).
NonMessageVariable	This error occurs when the source element is set to a variable type which is not a message.
InvalidXMLPayload	This error occurs when the input XML Payload is invalid.

Following fault variables is set when the policy triggers an error at runtime:

Fault Variables

Variable Set	Where	Example
[prefix].[policy_name].failed	The [prefix] is xmlattack. The [policy_name] is the name of the policy that threw the error.	xmlattack.XPT-SecureRequest.failed = true
fault.[error_name]	[error_name] = The specific error name to check for as listed in the table above.	fault.name Matches "SourceUnavailable"

Following is an example of an error response:

```

❏ Sample Code

{
 "fault": {
 "faultstring": "XMLThreatProtection[XPT-SecureRequest]: Execution failed.
reason: XMLThreatProtection[XTP-SecureRequest]: Exceeded object entry name
length at line 2",
 "detail": {
 "errorcode": "steps.xmlthreatprotection.ExecutionFailed"
 }
 }
}

```

```
}
```

Following is an example of a fault rule:

### Sample Code

```
<FaultRule name="XML Threat Protection Policy Faults">
 <Step>
 <Name>AM-CustomErrorResponse</Name>
 <Condition>(fault.name Matches "ExecutionFailed") </Condition>
 </Step>
 <Condition>(xmlattack.XPT-SecureRequest.failed = true) </Condition>
</FaultRule>
```

## Related Information

[JSON Threat Protection \[page 297\]](#)

[Regular Expression Protection \[page 414\]](#)

### 1.5.1.4.1.31 Regular Expression Protection

API Management helps to identify common content level threats that follow certain patterns, by enabling developers configure regular expressions that can be evaluated against API traffic at runtime.

This policy extracts information from a message (for example, URI Path, Query Param, Header, Form Param, Variable, XML Payload, or JSON Payload) and evaluates that content against predefined regular expressions. If any specified regular expressions evaluates to true, the message is considered a threat and is rejected. The most common usage of RegularExpressionProtection policy is the evaluation of payloads of JSON and XML for malicious content.

This policy supports regular expression rules as the classes in the java.util.regex package in java language.

An example payload for the policy is as follows:

### Code Syntax

```
<RegularExpressionProtection async="false" continueOnError="true"
enabled="true" xmlns="http://www.sap.com/apimgmt">
 <IgnoreUnresolvedVariables>>false</IgnoreUnresolvedVariables>

 <!--Validates if the Uri path has "internal" keywordd -->
 <URIPath>
 <Pattern>(.*)(internal)(.*)</Pattern>
 </URIPath>

 <!--Validates if the "action" query param has any sql
injection code to do any invasive operation -->
 <QueryParam name="action">
 <Pattern>[\s]*((delete)|(exec)|(drop\s*table)|(insert)|
(shutdown)|(update)|(\bor\b))</Pattern>
 </QueryParam>
```

```

 <!--Validates if the "action" header has any content with
"threat" string -->
 <Header name="action">
 <Pattern>(.*)(threat)(.*)</Pattern>
 </Header>
 <!--Validates if the "action" form param has any content with
"threat" string -->
 <FormParam name="action">
 <Pattern>(.*)(threat)(.*)</Pattern>
 </FormParam>
 <!--Validates if "flow.actionvar" variable has any content
with "threat" string -->
 <Variable name="flow.actionvar">
 <Pattern>(.*)(threat)(.*)</Pattern>
 </Variable>
 <Source>request</Source>
 <!--Validates if "command.action" node has any content with
"threat" string -->
 <JSONPayload>
 <JSONPath>
 <Expression>$.command.action</Expression>
 <Pattern>(.*)(threat)(.*)</Pattern>
 </JSONPath>
 </JSONPayload>
 <!--Validates if "/sap:Command/sap:Action" node has any
content with "threat" string -->
 <XMLPayload>
 <Namespaces>
 <Namespace prefix="sap">http://www.sap.com</
Namespace>
 </Namespaces>
 <XPath>
 <Expression>/sap:Command/sap:Action</Expression>
 <Type>string</Type>
 <Pattern>(.*)(threat)(.*)</Pattern>
 </XPath>
 </XMLPayload>
 </RegularExpressionProtection>

```

Attribute Name	Description	Presence
Source	<p>Indicates the message from which information needs to be extracted.</p> <p>If the &lt;Source&gt; element is omitted, the value defaults to message. For example, &lt;Source&gt;message&lt;/Source&gt;. When set to message, the policy uses the request message as source when attached to a request flow. Likewise, the policy uses the response message when attached to a response flow.</p> <p>If the source message cannot be resolved or if it resolves to a non-message type, the policy returns an error.</p> <p>&lt;Source&gt;response&lt;/Source&gt;</p>	Optional
IgnoreUnresolvedVariables	<p>If set to true and any variable is unresolvable, the policy returns an error and the unresolved variable is treated as empty string (Null).</p> <p>&lt;IgnoreUnresolvedVariables&gt;&gt;false&lt;/IgnoreUnresolvedVariables&gt;</p>	Optional

Attribute Name	Description	Presence
URIPath	Specifies the request URI path from where the information needs to be extracted and evaluated against the provided regular expression. Provide at least one <Pattern> element specifying a regular expression pattern to match.	Optional
	<pre> &lt;URIPath&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt; &lt;/URIPath&gt; </pre>	
QueryParam	Specifies the request query parameter from where the information needs to be extracted and evaluated against the provided regular expression. Provide at least one <Pattern> element specifying a regular expression pattern to match.	Optional
	<pre> &lt;QueryParam name="s-query-param"&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt; &lt;/QueryParam&gt; </pre>	
Header	Specifies the request and response header from where the information needs to be extracted and evaluated against the provided regular expression. Provide at least one <Pattern> element specifying a regular expression pattern to match.	Optional
	<pre> &lt;Header name="s-header"&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt; &lt;/Header&gt; </pre>	
FormParam	Specifies the request form parameter from where the information needs to be extracted and evaluated against the provided regular expression. Provide at least one <Pattern> element specifying a regular expression pattern to match.	Optional
	<pre> &lt;FormParam name="s-form-param"&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt; &lt;/FormParam&gt; </pre>	



Attribute Name	Description	Presence
Variable	Specifies the variable from where the information needs to be extracted and evaluated against the provided regular expression.	Optional
	<pre>&lt;Variable name="request.content"&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt; &lt;/Variable&gt;</pre>	
XMLPayload	Specifies theXML payload from where the information needs to be extracted and evaluated against the provided regular expression.	Optional
	<pre>&lt;XMLPayload&gt;   &lt;Namespaces&gt;    &lt;Namespace prefix="sap"&gt;http://   www.sap.com&lt;/Namespace&gt;   &lt;/Namespaces&gt;   &lt;XPath&gt;     &lt;Expression&gt;/sap:Greeting/     sap:User&lt;/Expression&gt;     &lt;Type&gt;string&lt;/Type&gt;     &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt;     &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt;   &lt;/XPath&gt; &lt;/XMLPayload&gt;</pre>	
JSONPayload	Specifies the JSON payload from where the information needs to be extracted and evaluated against the provided regular expression.	Optional
	<pre>&lt;JSONPayload&gt;   &lt;JSONPath&gt;    &lt;Expression&gt;\$.store.book[*].author&lt;/   Expression&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt;   &lt;Pattern&gt;REGEX PATTERN&lt;/   Pattern&gt;   &lt;/JSONPath&gt; &lt;/JSONPayload&gt;</pre>	

RegularExpressionProtection policy type defines the following error codes:

Error code	Message
NothingToEnforce	RegularExpressionProtection {0}: at least one of URIPath, QueryParam, Header, FormParam, XMLPayload, JSONPayload is mandatory
NoPatternsToEnforce	RegularExpressionProtection {0}: No patterns to enforce in {1}
EmptyXPathExpression	RegularExpressionProtection {0}: Empty XPath expression

Error code	Message
EmptyJSONPathExpression	RegularExpressionProtection {0}: Empty JSONPath expression
DuplicatePrefix	RegularExpressionProtection {0}: Duplicate prefix {1}
NONEmptyPrefixMappedToEmptyURI	RegularExpressionProtection {0}: Non-empty prefix {1} cannot be mapped to empty uri
ThreatDetected	Regular Expression Threat Detected in {0}; regex: {1} input: {2}
ExecutionFailed	Failed to execute the RegularExpressionProtection StepDefinition {0}. Reason: {1}
VariableResolutionFailed	Failed to resolve variable {0}
NonMessageVariable	Variable {0} does not resolve to a Message
SourceMessageNotAvailable	{0} message is not available for RegularExpressionProtection StepDefinition {1}
InvalidRegularExpression	RegularExpressionProtection {0}: Invalid Regular Expression {1}, Context {2}
InstantiationFailed	Failed to instantiate the RegularExpressionProtection StepDefinition {0}
CannotBeConvertedToNodeset	RegularExpressionProtection {0}: Result of xpath {1} cannot be converted to nodeset. Context {2}
XPathCompilationFailed	RegularExpressionProtection {0}: Failed to compile xpath {1}. Context {2}
JSONPathCompilationFailed	RegularExpressionProtection {0}: Failed to compile jsonpath {1}. Context {2}

### 1.5.1.4.1.32 Statistics Collector Policy

This policy enables you to collect statistics for data in a message, such as product ID, price, REST action, client and target URL, and message length.

The data comes from flow variables predefined by API Management or custom variables that you define. The statistics data is passed to the analytics server, which analyzes the statistics and generates reports. This can be viewed by creating custom charts.

Only one Statistics Collector policy should be attached to a single API proxy. If there are multiple Statistics Collector policies in a proxy, then the last one to execute determines the data written to the analytics server. You can attach the policy in one of the following locations:

ProxyEndpoint			TargetEndpoint			←Response	
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow		PostFlow
	•	•	•	•	•		•
	•	•	•	•	•		•
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload for the policy is as follows:

### Code Syntax

```
<!-- The policy collects data for each request and passes to the analytics
server, in the below policy,
the data is collected from productID and the price variables. if the variables
are not set, the default values are used -->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<StatisticsCollector xmlns="http://www.sap.com/apimgmt">
 <Statistics>
 <Statistic name="productID" ref="product.id" type="string">999999</
Statistic>
 <Statistic name="price" ref="product.price" type="string">0</
Statistic>
 </Statistics>
</StatisticsCollector>
```

### Note

If you attempt to apply a policy template that includes a Statistic Collector policy with the Statistic name='clientIP' to an API proxy, the application of the policy template will fail because 'clientIP' is a reserved word in Cloud Foundry API Management analytics. To address this issue, we have introduced a default prefix, "**custom\_**", which will be automatically appended to all custom metric names. For example, if the Statistic name is "clientIP", it will be displayed as "**custom\_clientIP**".

### Remember

If any changes are made to the existing policy, the API proxy must be redeployed.

The following table describes the attributes of the <Statistic> element:

Attribute	Default	Type	Description
name (Required)	N/A	N/A	<p>The name is used to reference the data collected for the specified variable. While viewing analytics data, use this name to reference the data collected about the variable specified by the <code>ref</code> attribute. If the variable specified by <code>ref</code> is undefined on a request or response, then <code>defaultStatValue</code> specifies the value collected for the variable. If you omit the default value, no data is collected for the variable when the variable is undefined.</p> <p>The following restrictions apply:</p> <ul style="list-style-type: none"> <li>Names cannot be multiple-word (no spaces).</li> <li>No spaces, m dashes, quotation marks, underscores, hyphens, or periods.</li> <li>No special characters.</li> </ul>
ref(Required)	N/A	N/A	<p>The flow variable for which you are collecting statistics. This variable can be a flow variable predefined by API Management or a custom variable that you define in your API proxy. The <code>ref</code> attribute often references a custom variable defined by the Extract Variables policy.</p>

Attribute	Default	Type	Description
type (Optional)	N/A	String/Integer/Float/Long/ Double	<p>Specifies the data type of the variable specified by the <code>ref</code> attribute.</p> <p>For data of type String, reference the statistical data as a Dimension in a custom report. For numerical data types (integer/float/long/double), reference the statistical data in a custom report as a Metric. The value of type can be omitted only if <code>ref</code> refers to a predefined API Management flow variable or the <code>type</code> is declared in the XML payload of the Extract Variables policy.</p>

During the policy execution, the following errors can occur:

Deployment errors

Error Name	Fault String	Cause
UnsupportedDatatype	StatisticsCollection [collection]: Datatype [type] is unsupported. Context [context]	If the type of the variable specified by the <code>ref</code> attribute in the <code>&lt;Statistic&gt;</code> element of the Statistics Collector policy is unsupported, then the deployment of the API proxy fails. The supported data types are string, integer, float, long, double, and boolean.
InvalidName	StatisticsCollection: Name: [name] conflicts with system defined variables. Context [context]	If the name used to reference the data collected for the specified variable defined within the <code>&lt;Statistic&gt;</code> element of the Statistics Collector policy conflicts with a system-defined variable, then the deployment of the API proxy fails. Some of the known system-defined variables are organization and environment.
DatatypeMissing	StatisticsCollection [name]: Datatype of [type] is missing. Context [context]	If the type of the variable specified by the <code>ref</code> attribute in the <code>&lt;Statistic&gt;</code> element of the Statistics Collector policy is missing, then the deployment of the API proxy fails.

## 1.5.1.4.1.33 Open Connectors

API management provides open connector policy to be attached to an Open Connector type API.

For an open connector type API, you can attach only one open connector policy. The policy is either attached to the target endpoint or the proxy endpoint.

You can attach the policy in the following locations:

ProxyEndpoint			TargetEndpoint			←Response	
Request →	PreFlow	Flow	PostFlow	PreFlow	Flow		PostFlow
	•	•	•	•	•		•
	•	•	•	•	•		•
	PostFlow	Flow	PreFlow	PostFlow	Flow	PreFlow	

An example payload of the payload is as follows:

### Sample Code

```
<OpenConnectors async="true" continueOnError="false"
enabled="true" xmlns="http://www.sap.com/apimgmt">
 <InstanceSecret kvm-map-name="apim.oc.instance.token" kvm-key-
name="default"></InstanceSecret>
</OpenConnectors>
For API proxies that are created via service or imported,
before attaching the policy, ensure that there is a KVM created with map name
same as 'kvm-map-name' and KVM key with 'kvm-key-name'.
```

Following table lists the elements and attributes that you can configure on this policy.

Elements and Attributes	Description
InstanceSecret	Specifies the KVM map name and key.
kvm-map-name	Map name that specifies the instance token.
kvm-key-name	Specifies the key to the KVM.

## Open Connector Callout

This section describes the process to add a service callout policy to an open connector type API.

In the policy editor window, choose open connector policy. In the create policy window, enter the required details. Select the *External Call* checkbox. From the API Provider dropdown, discover an API Provider (Only open connector type Providers are listed here)).

An example payload of the payload is as follows:

### Sample Code

```

 <!-- Environment and Proxy scoped variables will be retrived and
 passed as authorization header to Backend -->
 <OpenConnectorCallout xmlns="http://www.sap.com/apimgmt" async="true"
 continueOnError="false" enabled="true">
 <Request>
 <Set>
 <Headers></Headers>
 <FormParams></FormParams>
 <QueryParams></QueryParams>
 <Path>/elements/api-v2/ping</Path>
 <Verb>GET</Verb>
 <Payload/>
 </Set>
 </Request>
 <APIProvider>OcSanity</APIProvider>
 <Timeout>30000</Timeout>
 <!-- PUT stores the key value pair mentioned inside the element
 -->
 <InstanceSecret kvm-map-
 name="apim.occallout.instance.secret.Test5512" kvm-key-name="Haha_MyGmail"/>
 <Response/>
 </OpenConnectorCallout>
 # For API proxies that are created via service or imported, before
 attaching the policy, ensure that there is a KVM created with map name same
 as 'kvm-map-name' and KVM key with 'kvm-key-name'.

```

Following table lists the elements and attributes that you can configure on this policy.

Elements and Attributes	Description
APProvider	Name of the open connector typeAPI Provider.
Timeout	Time the policy waits for a response from the target. Time in milliseconds
Headers	Overwriting existing HTTP headers. <div data-bbox="683 1245 1394 1503" data-label="Code-Block"> <p>↔ Sample Code</p> <p>This example sets the 'test' header to 'request.header.test'.</p> <pre> &lt;Headers&gt;     &lt;Header     name="test"&gt;{request.header.test}&lt;/Header&gt; &lt;/Headers&gt; </pre> </div>
QueryParams	Adds new query parameters to the request. Use this attribute for GET operation only. <div data-bbox="683 1603 1394 1917" data-label="Code-Block"> <p>↔ Sample Code</p> <p>The following example sets the "address" query parameter to the value of the request.header.address variable:</p> <pre> &lt;QueryParams&gt;     &lt;QueryParam     name="address"&gt;{request.header.address}&lt;/     QueryParam&gt; &lt;/QueryParams&gt; </pre> </div>
Path	Path to the endpoint that is targeted

Elements and Attributes	Description
InstanceSecret	Specifies the KVM map name and key.
kvm-map-name	Map name that specifies the instance token.
kvm-key-name	Specifies the key to the KVM.

## 1.5.1.4.2 Variable References

describes a set of variables predefined by the API Services.

Types of variables available are:

- [Request Message Variables \[page 438\]](#)
- [System Variables \[page 451\]](#)
- [Response Message Variables \[page 459\]](#)
- [Message Variables \[page 457\]](#)
- [Error Variables \[page 454\]](#)
- [Configuration Variables \[page 453\]](#)
- [Path Variables \[page 463\]](#)
- [API Proxy Flows Variables \[page 456\]](#)
- [TLS/SSL Connection Information Variables \[page 454\]](#)



## 1.5.1.4.2.1 Flow Variables

This topic provides information about the flow variables.

### Client Flow Variables

Variable	Description	Scope begins	Type	Permission
client.cn	The common name specified in the TLS/SSL certificate presented by the client app.	Proxy request	String	Read

#### ⓘ Note

If the common name in the TLS/SSL certificate contains comma separated values, then the client.cn flow variable returns only the first value before comma. If you need to read all the values, then it is recommended to use the tls.client.s.dn flow variable. For more information, see [TLS/SSL Connection Information Variables \[page 454\]](#).

This behaviour applies to all the client flow variables that are associated with TLS/SSL context. For example, client.country, client.email.address, client.organization, client.organization.unit, cli-

Variable	Description	Scope begins	Type	Permission
	ent.locality, and client.state.			
client.country	The country/region in the TLS/SSL certificate presented by the client app.	Proxy request	String	Read
client.email.address	The e-mail address in the TLS/SSL certificate presented by the client app.	Proxy request	String	Read
client.host	The HTTP host IP associated with the request received by the ProxyEndpoint.	Proxy request	String	Read
client.ip	The IP address of the client or system sending the message. For example, this could be the original client IP or a load balancer IP.	Proxy request	String	Read
client.locality	The locality (City) in the TLS/SSL certificate presented by the client.	Proxy request	String	Read
client.organization	The organization in the TLS/SSL certificate presented by the client.	Proxy request	String	Read
client.organization.unit	The organizational unit in the TLS/SSL certificate presented by the client.	Proxy request	String	Read
client.port	The HTTP port associated with the originating client request to ProxyEndpoint.	Proxy request	Integer	Read

Variable	Description	Scope begins	Type	Permission
client.received.end.time	<ul style="list-style-type: none"> <li>The time, expressed in string form, at which the proxy finished receiving the request from the originating client at the ProxyEndpoint. For example: Wed, 21 Aug 2013 19:16:47 UTC</li> <li>This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.</li> </ul>	Proxy request	String	Read
client.received.end.timestamp	The timestamp value specifying when the proxy finished receiving the request from the originating client at the ProxyEndpoint. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Proxy request	Long	Read

Variable	Description	Scope begins	Type	Permission
client.received.start.time	<ul style="list-style-type: none"> <li>The time, expressed in string form, at which the proxy began receiving the request from the originating client at the ProxyEndpoint. For example: Wed, 21 Aug 2013 19:16:47 UTC</li> <li>This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.</li> </ul>	Proxy request	String	Read
client.received.start.timestamp	The timestamp value specifying when the proxy began receiving the request from the originating client at the ProxyEndpoint. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Proxy request	Long	Read

Variable	Description	Scope begins	Type	Permission
client.sent.end.time	<ul style="list-style-type: none"> <li>The time, expressed in string form, when the ProxyEndpoint finished returning the response to the originating client app. For example: Wed, 21 Aug 2013 19:16:47 UTC</li> <li>This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.</li> </ul>	Proxy response	String	Read
client.sent.end.time-stamp	The timestamp value specifying when the ProxyEndpoint finished returning the response to the originating client app. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Proxy response	Long	Read

Variable	Description	Scope begins	Type	Permission
client.sent.start.time	<ul style="list-style-type: none"> <li>The time, expressed in string form, when the ProxyEndpoint started returning the response to the originating client app. For example: Wed, 21 Aug 2013 19:16:47 UTC</li> <li>This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.</li> </ul>	Proxy response	String	Read
client.sent.start.time-stamp	The timestamp value specifying when the ProxyEndpoint started returning the response to the originating client app. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Proxy response	Long	Read
client.scheme	Returns http or https depending on the transport used by client app to send the request message.	Proxy request	String	Read
client.state	The state in the TLS/SSL certificate presented by the client.	Proxy request	String	Read
client.ssl.enabled	Returns true or false, depending on whether the ProxyEndpoint is configured for TLS/SSL.	Proxy request	String	Read
onpremise.target.host	The onpremise virtual host configured in onpremise APIProvider	Proxy target endpoint request	String	Read

Variable	Description	Scope begins	Type	Permission
onpremise.target.port	The onpremise virtual port configured in onpremise APIProvider	Proxy target endpoint request	String	Read
onpremise.target.basepath	Determines the actual base path.	Proxy target endpoint request	String	Read

#### Error Flow Variables

Variable	Description	Scope	Type	Permission
error	Error of type Message, which is a contextual object in the error flow	Error	Message	Read/Write
error.content	Content of the error	Error	String	Read/Write
error.message	Message associated with an error, whose value is available only before the error Flow is executed.	Error	String	Read
error.status.code	The HTTP status code associated with the error. For example: "400".	Error	String	Read
error.reason.phrase	The reason phrase associated with the error. For example: "Bad Request"	Error	String	Read
error.transport.message	Any error of type TransportMessage	Error	Transport_Message	Read
error.state	State in the Flow where an error occurred	Error	Integer	Read
error.header.<name>	Get or set the response header.	Error	integer	Read/Write

#### Request Flow Variables

Variable	Description	Scope	Type	Permission
request	The complete request, including any payload present.	Proxy request	Message	Read
request.content	Gets or sets the payload of the request message.	Proxy request	String	Read/Write
request.formparam.{formparam_name}	Gets or sets the value of the specified form parameter in the request sent from the client app.	Proxy request	String	Read/Write

Variable	Description	Scope	Type	Permission
request.formparam.{formparam_name}.values.count	Count of all values for the specified form parameter associated with the request.	Proxy request	Integer	Read
request.formparams.count	Count of all form parameters associated with the request sent from the client app.	Proxy request	Integer	Read
request.formparams.names	A list of all form parameter names associated with the request.	Proxy request	Collection	Read
request.header.{header_name}	Gets or sets the value of a particular header found in the request.	Proxy request	String	Read/Write
<div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p><b>Note</b></p> <p>If the {header_name} has multiple values, you can read or retrieve all the values of the header. For more information on extracting multiple values of an HTTP header, see <a href="#">Multi-value HTTP Headers in an API Proxy [page 464]</a></p> </div>				
request.header.{header_name}.values	All the values of a particular header in the request	Proxy request	Collection	Read
request.header.{header_name}.values.count	Count of all the values of a particular header in the request.	Proxy request	Integer	Read
request.formparams.count	Count of all form parameters associated with the request sent from the client a	Proxy request	Integer	Read
request.formparams.names	A list of all form parameter names associated with the request.	Proxy request	Collection	Read
request.header.{header_name}	Gets or sets the value of a particular header found in the request.	Proxy request	String	Read/Write



Variable	Description	Scope	Type	Permission
request.header. {header_name}.values	All the values of a particular header in the request	Proxy request	String	Read
request.header. {header_name}.values.count	Count of all the values of a particular header in the request.	Proxy request	Integer	Read
request.headers.count	Count of all the headers in the request.	Proxy request	Integer	Read
request.path	The unproxied resource path (not including the host) to the backend service, excluding query parameters.	Proxy request	Integer	Read
request.query-param. {query-param_name}	The value of a particular query parameter found in the request.	Proxy request	string	Read/Write
request.query-param. {query-param_name}.values.count	The count of all the values of a particular query parameter in the request.	Proxy request	Integer	Read
request.queryparams.count	The count of all the query parameters in the request.	Proxy request	Integer	Read
request.queryparams.names	request.queryparams.names	Proxy request (Java Object)	Collection	Read
request.querystring	The complete list of query parameters in the request sent from the client app. For example, if the request is <b>http://host.com/123?name=first&amp;surname=second&amp;place=address</b> then this variable returns name=first&surname=second&place=address	Proxy request	String	Read
request.transportid	ID of the request as type TransportMessage which is a contextual object	Proxy request	String	Read
request.transport.message	Request of type TransportMessage which is a contextual object	Proxy request	Transport	Read

Variable	Description	Scope	Type	Permission
request.uri	<p>In an API proxy, the proxy &lt;BasePath&gt; in the ProxyEndpoint (in addition to the proxy's base URL) maps to the target service URL in the TargetEndpoint. For example:</p> <pre>&lt;ProxyEndpoint&gt; ... &lt;BasePath&gt;/my- mock-proxy&lt;/Base- Path&gt; points to &lt;TargetEndpoint&gt; ... &lt;HTTPTargetConne- ction&gt; http://mocktar- get.sap.com &lt;/HTTPTargetConne- ction&gt;</pre> <p>In the request, the request.uri is the proxy base path + the remainder of the address, including query parameters.</p> <p>In the response, the request.uri is the remainder of the address, including query parameters, after the HTTPTargetConnection.</p> <p>The difference is because the original request came into the proxy, but then the proxy makes another</p>	Proxy request (differs with response)	String	Read

Variable	Description	Scope	Type	Permission
	<p>request to the target service.</p> <p>Let's say the following call is made to our sample proxy, which has a base path of /my-mock-proxy:</p> <p>http://my_org-test.sap.com/my-mock-proxy/user?user=test</p> <p>and the proxy calls</p> <p>http://mocktarget.apigee.net (which appends /user?user=test to that URL).</p> <p>Request: request.uri = /my-mock-proxy/user?user=test</p> <p>Response: request.uri = /user?user=test</p>			
request.url	Returns the exact complete URL of the final request made.	Target request	String	Read
request.verb	The HTTP verb used for the request. For example: GET, PUT, DELETE	Proxy request	String	Read
request.version	Gets the HTTP version of the request.	Proxy request	String	Read
response.formstring	<p>The complete list of form parameters in the request.</p> <p>For example: name=test&amp;type=first &amp;group=A</p>	Target request	String	Read

Variable	Description	Scope	Type	Permission
route.name	The name of the RouteRule that was executed in the ProxyEndpoint. For example: default. A RouteRule references an API proxy TargetEndpoint to execute.	Target request	String	Read
route.target	The name of the TargetEndpoint that was executed. For example: default.	Target request	String	Read

#### Response Flow Variables

Variable	Description	Scope begins	Type	Permission
response	Complete response message returned by target	Target response	Message	Read/Write
response.content	Payload content of the response message returned by the target	Target response	String	Read/Write
response.formparam.{formparam_name}	The value of a form parameter in the response	Target response	String	Read/Write
response.formparam.{formparam_name}.values.count	Count of all the values of the specified form parameter in response	Target response	Integer	Read
response.formparams.count	Count of all form parameters in the response	Target response	Integer	Read
response.formparams.names	The names of all the form parameters in the response	Target response	Collection	Read

Variable	Description	Scope begins	Type	Permission
response.header. {header_name}	Gets or sets the value of a specified HTTP header in the response. If the header has multiple values (such as a CSV list), a GET returns the first value only.	Target response	String	Read/Write
<div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p><b>Note</b></p> <p>To read or retrieve all the values of the header, you can see <a href="#">Multi-value HTTP Headers in an API Proxy [page 464]</a></p> </div>				
response.header. {header_name}.values	All the values of a specified HTTP header in response.	Target response	String	Read
response.header. {header_name}.values.count	Count of all the values of the specified HTTP header in response	Target response	Integer	Read
response.headers.count	Count of all the headers in the response	Target response	Integer	Read
response.headers.names	The names of all the headers in the response	Target response	Integer	Read
response.reason.phrase	The response reason phrase for a particular request	Target response	String	Read
response.status.code	The response code returned for a request. You can use this variable to override the response status code, which is stored in message.status.code.	Target response	Integer	Read/Write
response.transport.message	Response of type TransportMessage which is a contextual object	Target response	String	Read

## 1.5.1.4.2 Request Message Variables

Request message variables are used in policies to access message components like the header, the query parameters, form parameters, the source IP address, the HTTP message body.

API proxy applies the incoming request to a series of policies, depending on the request condition, API proxy can either modify or transform the request. Based on the content of the request variable, policies can either transform or reject the request. Supported request variables are listed in the Request Message Variables table.

Request Message Variables

Variable	Description	Scope	Type	Permission
client.host	The HTTP host IP associated with the request received by the ProxyEndpoint.	Proxy request	String	Read
client.ip	The IP address of the client or system sending the message to the Edge router. For example, this could be the original client IP or a load balancer IP.	Proxy request	String	Read
client.port	The HTTP port associated with the originating client request to ProxyEndpoint.	Proxy request	Integer	Read
client.received.end.timestamp	The timestamp value specifying when the proxy finished receiving the request from the originating client at the ProxyEndpoint. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Proxy request	Long	Read

Variable	Description	Scope	Type	Permission
client.received.start.time	<p>The time, expressed in string form, at which the proxy began receiving the request from the originating client at the ProxyEndpoint. For example: Wed, 21 Aug 2013 19:16:47 UTC</p> <p>This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.</p>	Proxy request	String	Read
client.received.start.timestamp	<p>The timestamp value specifying when the proxy began receiving the request from the originating client at the ProxyEndpoint. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.</p>	Proxy request	Long	Read

Variable	Description	Scope	Type	Permission
client.sent.end.time	<p>The time, expressed in string form, when the ProxyEndpoint finished returning the response to the originating client app. For example: Wed, 21 Aug 2013 19:16:47 UTC</p> <p>This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.</p>	Proxy request	String	Read
client.sent.end.time-stamp	<p>The timestamp value specifying when the ProxyEndpoint finished returning the response to the originating client app. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.</p>	Proxy request	Long	Read



Variable	Description	Scope	Type	Permission
client.sent.start.time	<p>The time, expressed in string form, when the ProxyEndpoint started returning the response to the originating client app. For example: Wed, 21 Aug 2013 19:16:47 UTC</p> <p>This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.</p>	Proxy request	String	Read
client.sent.start.time-stamp	The timestamp value specifying when the ProxyEndpoint started returning the response to the originating client app. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Proxy request	Long	Read
client.scheme	Returns http or https depending on the transport used by client app to send the request message.	Proxy request	String	Read
message.queryparam.{query-param_name}	Returns the specified message query parameter.	Proxy request	String	Read
message.queryparam.{query-param_name}.values.count	The total count of a specified query parameter associated with the request sent to the ProxyEndpoint from the client app	Proxy request	Integer	Read

Variable	Description	Scope	Type	Permission
message.queryparams.count	The total count of all query parameters associated with the request sent to the ProxyEndpoint from the client app.	Proxy request	Integer	Read
message.queryparams.names	A list of all query parameter names associated with the request sent to the ProxyEndpoint from the client app.	Proxy request	Collection (Java Object)	Read
message.querystring	A string containing all query parameter names and values associated with the request sent to the ProxyEndpoint from the client app.	Proxy request	String	Read
message.uri	The complete URI path (following the domain URL) including query parameters.	Proxy request	String	Read
message.verb	The HTTP verb (GET, PUT, POST, DELETE, and so on) associated with the request	Proxy request	String	Read
message.version	The HTTP version associated with the request sent to the ProxyEndpoint from the client app.	Proxy request	String	Read/Write
messageid	This ID is logged in the error logs to correlate the messageid with the errors.	Proxy request	String	Read
proxy.basepath	The value of the Base Path in your API proxy configuration. The base path is the URI fragment that follows the host in the URL. Conditional flow URIs follow the base path.	Proxy request	String	Read

Variable	Description	Scope	Type	Permission
proxy.client.ip	The <b>X-Forwarded-For</b> address of the inbound call, which is the IP address API Management received from the last external TCP handshake. This could be the calling client.	Proxy request	String	Read
proxy.pathsuffix	<p>The value of API proxy basepath suffix that is sent from the client and received at the ProxyEndpoint.</p> <p>The basepath is defined as the path component that uniquely identifies the API proxy. The public-facing URL of an API proxy is comprised of your organization name, the environment where the proxy is deployed, the basepath, the basepath suffix, and any query parameters.</p> <p>For example, in the following request:</p> <pre>http://myorg-test.sap.net/v2/weatherapi/forecastrss?w=12797282</pre> <p>The basepath suffix is /forecastrss</p>	Proxy request	String	Read
proxy.url	Gets the complete URL associated with the proxy request received by the ProxyEndpoint, including any query parameters present. Note that the host in proxy.url is the router host, not the host used in the original request.	Proxy request	String	Read

Variable	Description	Scope	Type	Permission
request	The complete request, including any payload present.	Proxy request	Message	Read
request.content	Gets or sets the payload of the request message.	Proxy request	String	Read/Write
request.for- mparam.{for- mparam_name}	Gets or sets the value of the specified form parameter in the request sent from the client app.	Proxy request	String	Read/Write
request.for- mparam.{for- mparam_name}. values.count	Count of all values for the specified form parameter associated with the request.	Proxy request	Integer	Read
request.formpar- ams.count	Count of all form parameters associated with the request sent from the client app.	Proxy request	Integer	Read
request.formpar- ams.names	A list of all form parameter names associated with the request.	Proxy request	Collection	Read
request.header. {header_name}	Gets or sets the value of a particular header found in the request.	Proxy request	String	Read/Write
request.header. {header_name}.values	All the values of a particular header in the request	Proxy request	Collection	Read
request.header. {header_name}.val- ues.count	Count of all the values of a particular header in the request.	Proxy request	Integer	Read
request.headers.count	Count of all the headers in the request.	Proxy request	Integer	Read
request.path	The un-proxied resource path (not including the host) to the backend service, excluding query parameters.	Proxy request	String	Read
request.query- param.{query- param_name}	The value of a particular query parameter found in the request.	Proxy request	String	Read/Write
request.query- param.{query- param_name}.val- ues.count	The count of all the values of a particular query parameter in the request.	Proxy request	Integer	Read

Variable	Description	Scope	Type	Permission
request.queryparams.count	The count of all the query parameters in the request.	Proxy request	Integer	Read
request.queryparams.names	The names of all the query parameters in the request.	Proxy request	Collection (JavaObject)	Read
request.querystring	<p>The complete list of query parameters in the request sent from the client app.</p> <p>For example, if the request is</p> <p>http://host.com/123?name=first&amp;surname=second&amp;place=address</p> <p>then this variable returns</p> <p>name=first&amp;surname=second&amp;place=address</p>	Proxy request	String	Read
request.transportid	ID of the request as type TransportMessage which is a contextual object	Proxy request	String	Read
request.transport.message	Request of type TransportMessage which is a contextual object	Proxy request	Transport message	Read

Variable	Description	Scope	Type	Permission
request.uri	<p>In an API proxy, the proxy &lt;BasePath&gt; in the ProxyEndpoint (in addition to the proxy's base URL) maps to the target service URL in the TargetEndpoint. For example:</p> <pre> &lt;ProxyEndpoint&gt; ... &lt;BasePath&gt;/my-mock-proxy&lt;/BasePath&gt; points to &lt;TargetEndpoint&gt; ... &lt;HTTPTargetConnection&gt; http://mocktarget.sap.com &lt;/HTTPTargetConnection&gt; </pre> <p>In the request, the request.uri is the proxy base path + the remainder of the address, including query parameters.</p> <p>In the response, the request.uri is the remainder of the address, including query parameters, after the HTTPTargetConnection.</p> <p>The difference is because the original request came into the proxy, but then the proxy makes another</p>	Proxy request	String	Read

Variable	Description	Scope	Type	Permission
	<p>request to the target service.</p> <p>Let's say the following call is made to our sample proxy, which has a base path of /my-mock-proxy:</p> <p>http://my_org-test.sap.com/my-mock-proxy/user?user=test</p> <p>and the proxy calls</p> <p>http://mocktarget.apigee.net (which appends /user?user=test to that URL).</p> <p>Request: request.uri = /my-mock-proxy/user?user=test</p> <p>Response: request.uri = /user?user=test</p>			
request.url	Returns the exact complete URL of the final request made.	Target request	String	Read
request.verb	The HTTP verb used for the request. For example: GET, PUT, DELETE	Proxy request	String	Read
request.version	Gets the HTTP version of the request.	Proxy request	String	Read
response.formstring	<p>The complete list of form parameters in the request.</p> <p>For example: name=test&amp;type=first &amp;group=A</p>	Target request	String	Read

Variable	Description	Scope	Type	Permission
route.name	The name of the RouteRule that was executed in the ProxyEndpoint. For example: default. A RouteRule references an API proxy TargetEndpoint to execute.	Target request	String	Read
route.target	The name of the TargetEndpoint that was executed. For example: default.	Target request	String	Read
servicecallout.requesturi	The TargetEndpoint URI for a ServiceCallout policy. The URI is the TargetEndpoint URL without the protocol and domain specification.	Proxy request	String	Read/Write
servicecallout.{policy-name}.expectedcn	The expected Common Name of the TargetEndpoint as referred to in a ServiceCallout policy. This is meaningful only when the TargetEndpoint refers to an TLS/SSL endpoint.	Proxy request	String	Read/Write
servicecallout.{policy-name}.target.url	The TargetEndpoint URL for a particular ServiceCallout policy.	Proxy request	String	Read/Write
target.basepath	Returns basepath of TargetEndpoint	Target Request	String	Read
target.copy.pathsuffix	When true, request forwarded from ProxyEndpoint to TargetEndpoint retains path suffix (the URI path fragment following the URI defined in the ProxyEndpoint base path.)	Target Request	Boolean	Read/Write
target.copy.queryparams	When true, request forwarded from ProxyEndpoint to TargetEndpoint retains query parameters.	Target Request	Boolean	Read/Write



Variable	Description	Scope	Type	Permission
target.cn	The Common Name of the TargetEndpoint. This is meaningful only when the TargetEndpoint refers to an TLS/SSL endpoint.	Target request	Boolean	Read
target.expectedcn	The expected Common Name of the TargetEndpoint. This is meaningful only when the TargetEndpoint refers to an TLS/SSL endpoint.	Proxy Request	String	Read
target.name	Target to which message is reaching from targetendpoint	Target Request	String	Read
target.sent.end.time	<p>The time, expressed in string form, at which the proxy stopped sending the request to the URL specified in the TargetEndpoint. For example: Wed, 21 Aug 2013 19:16:47 UTC</p> <p>This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.</p>	Target Request	String	Read
target.sent.end.time-stamp	The timestamp value specifying when the proxy finished sending the request to the URL specified in the TargetEndpoint. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Target Request	Long	Read

Variable	Description	Scope	Type	Permission
target.sent.start.time	<p>The time, expressed in string form, at which the proxy began sending the request to the URL specified in the TargetEndpoint. For example: Wed, 21 Aug 2013 19:16:47 UTC</p> <p>This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.</p>	Target Request	String	Read
target.sent.start.timestamp	The timestamp value specifying when the proxy started sending the request to the URL specified in the TargetEndpoint. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Target Request	Long	Read
target.ssl.enabled	Whether TargetEndpoint is running on TLS/SSL	Target Request	Boolean	Read
target.url	The URL configured in the TargetEndpoint XML file or the dynamic target URL (if target.url is set during the message flow). The variable does not include any additional path elements or query parameters. Returns null if called out of scope or otherwise unset.	Target Request	String	Read/Write

Variable	Description	Scope	Type	Permission
variable.expectedcn	Variable exposed for the common name if it's running on TLS/SSL	Proxy Request	String	Read/Write
virtualhost.aliases	Host aliases of the virtual host that is hit during a particular request	Proxy Request	String_array	Read
virtualhost.name	Name of the virtual host that serves the originating client request	Proxy Request	String	Read
virtualhost.ssl.enabled	Returns true if TLS/SSL is enabled in the virtual host configuration	Proxy request	Boolean	Read

### 1.5.1.4.2.3 System Variables

Information pertaining to the system is described in system variables.

Every system variable consists of two parts, a prefix `_system` and a function. For example: `system.time`, `system` is the prefix and `time` is the function. Supported system variables are listed in the System Variables table.

System Variables

Variables	Description	Scope	Type	Permission
system.timestamp	The timestamp value specifying when the request is received from the client at the ProxyEndpoint. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Proxy Request	Long	Read

Variables	Description	Scope	Type	Permission
system.time	<p>The time, expressed in string form, at which the proxy received a request from a client at the ProxyEndpoint. For example: Wed, 21 Aug 2013 19:16:47 UTC.</p> <p>This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.</p>	Proxy Request	String	Read
system.time.year	The year portion of the system.time variable.	Proxy request	Integer	Read
system.time.month	The month portion of the system.time variable.	Proxy request	Integer	Read
system.time.day	The day of month portion of the system.time variable.	Proxy request	Integer	Read
system.time.dayof-week	The day of the week portion of the system.time variable.	Proxy request	Integer	Read
system.time.hour	The hour portion of the system.time variable.	Proxy request	Integer	Read
system.time.minute	The minute portion of the system.time variable.	Proxy request	Integer	Read
system.time.second	The second portion of the system.time variable.	Proxy request	Integer	Read
system.time.millisecond	The millisecond portion of the system.time variable.	Proxy request	Integer	Read
system.time.zone	Timezone of the system.	Proxy request	String	Read
system.interface.{interface_name}	IP Address of the system.	Proxy request	String	Read

Variables	Description	Scope	Type	Permission
system.pod.name	The name of the pod where the proxy is running.	Proxy request	String	Read
system.region.name	The name of the data center region where the proxy is running.	Proxy request	String	Read
system.uuid	The UUID of the message processor handling the proxy.	Proxy request	String	Read
router.uuid	The UUID of the router handling the proxy	Proxy request	String	Read

## 1.5.1.4.2.4 Configuration Variables

Configuration Variables describes the configuration settings.

Supported configuration variables are listed in the Configuration Variables table.

Configuration Variables

Variables	Description	Scope	Type	Permission
organization.name	Name of the organization	Proxy request	String	Read
environment	Container for environment.name.	Proxy request	String	Read
environment.name	Name of the environment in which the transaction ran	Proxy request	String	Read
apiproxy.name	Name of the api proxy	Proxy request	String	Read
apiproxy.revision	The revision number of an API proxy	Proxy request	String	Read
is.error	Error flag	Proxy request	Boolean	Read
proxy.name	The name attribute configured for the ProxyEndpoint	Proxy request	String	Read

## 1.5.1.4.2.5 Error Variables

Supported error variables are listed in the Error Variables table.

Error Variables

Variable	Description	Scope	Type	Permission
error	Error of type Message, which is a contextual object in the error flow	Error	Message	Read/Write
error.content	Content of the error	Error	String	Read/Write
error.message	Message associated with an error, whose value is available only before the error Flow is executed.	Error	String	Read
error.status.code	The HTTP status code associated with the error. For example: "400".	Error	String	Read
error.reason.phrase	The reason phrase associated with the error. For example: "Bad Request".	Error	String	Read
error.transport.message	Any error of type TransportMessage	Error	Transport_Message	Read
error.state	State in the Flow where an error occurred	Error	Integer	Read
error.header.<name>	Get or set the response header.	Error	String	Read/Write

## 1.5.1.4.2.6 TLS/SSL Connection Information Variables

lets you enable one-way and two-way TLS/SSL support for virtual hosts.

When you access an API proxy through a virtual host that supports TLS/SSL, API Management captures information about the TLS connection. You can then access the TLS connection information in an API proxy through flow variables.

The kind of TLS/SSL information captured depends upon whether the virtual host is enabled for one-way or two-way TLS. For example, for one-way TLS, API Management captures information about TLS cipher or TLS protocol used in the TLS connection. For two-way TLS, API Management not only captures the same information as captured for one-way TLS, but also captures information about the client's certificate (cert). For example, the subject or issuer DN of the client cert, the serial number of the client cert and the client cert in the PEM format.

The following are the list of flow variables that contain TLS connection information and are available for access in the API proxy:

### Note

The following TLS information is captured for both one-way and two-way TLS when `<ConnectionProperties>` is set to true in the virtual host configuration file.

Variable	Description
tls.cipher	The cipher used by the TLS/SSL connection.
tls.protocol	The protocol used by the TLS/SSL connection.
tls.server.name	The requested SNI server name.
tls.session.id	The session identifier.  This flow variable is available when you set either <code>&lt;ConnectionProperties&gt;</code> or <code>&lt;ClientProperties&gt;</code> to true.

The following are the list of flow variables that contain TSL connection information pertaining to the client's cert.

### Note

The following TLS information is captured for two-way TLS when `<ClientProperties>` is set to true in the virtual host configuration file.

Variable	Description
tls.client.s.dn	The subject Distinguished Name (DN) of the client cert. This variable enables you to capture information about the subject (individual) being certified, including common name (client.cn), organization (client.organization), organization unit (client.organization.unit), e-mail address (client.email.address), country/region codes (client.country), locality (client.locality) etc.
tls.client.i.dn	The issuer Distinguished Name (DN) of the client cert.
tls.client.raw.cert	The client cert in the PEM format.
tls.client.cert.serial	The serial number of the client cert.
tls.client.cert.fingerprint	The SHA1 fingerprint of the client cert.
tls.session.id	The session identifier.  This flow variable is available when you set either <code>&lt;ConnectionProperties&gt;</code> or <code>&lt;ClientProperties&gt;</code> to true.

To configure a virtual host to capture the TLS/SSL information, you need to request the operations team to set the following properties to true in the virtual host configuration file:

Virtual Host Property	Description
ConnectionProperties	Set it to true to capture TLS connection information for both one-way and two-way TLS.
ClientProperties	Set it to true to capture additional information for two-way TLS.

## Related Information

[Flow Variables \[page 425\]](#)

### 1.5.1.4.2.7 API Proxy Flows Variables

Supported API proxy flow variables are listed in the API proxy flow Variables table.

API proxy flows variables

Variable	Description	Scope	Type	Permission
current.flow.name	The name of the currently executed flow (such as "PreFlow", "PostFlow", or the name of a conditional flow).	Proxy request	String	Read
current.flow.description	The description of the currently executed flow.	Proxy request	String	Read
proxy.flow.name	The name of the most recently executed ProxyEndpoint flow (such as "PreFlow", "PostFlow", or the name of a conditional flow).	Proxy request	String	Read
proxy.flow.description	The description of the most recently executed ProxyEndpoint flow.	Proxy request	String	Read
target.flow.name	The name of the most recently executed TargetEndpoint flow (such as "PreFlow", "PostFlow", or the name of a conditional flow).	Proxy request	String	Read



Variable	Description	Scope	Type	Permission
target.flow.description	The description of the most recently executed TargetEndpoint flow.	Proxy request	String	Read

### 1.5.1.4.2.8 Message Variables

Message variables refer to different message types like request, response, or error depending on the point within the APIproxy flow in which they are called.

Supported message variables are listed in the Message Variables table.

Message Variables

Variable	Description	Scope	Type	Permission
message	A contextual object, with the same value as request in the request Flow or as response in the response Flow or as error in the Error flow.	Proxy request	Message	Read/Write
message.content	Content of the request, response, or error message	Proxy request	String	Read/Write
message.formparam.{formparam_name}	Value of the specified form parameter	Proxy request	String	Read/Write
message.formparam.{formparam_name}.values	All values of the specified form parameter in the message	Proxy request	Collection	Read
message.formparam.{formparam_name}.values.count	Count of the values of the specified form parameters in the message	Proxy request	Integer	Read
message.formparams.count	Count of all form parameters in the message	Proxy request	Integer	Read
message.formparams.names	Value of all form parameters in the message	Proxy request	Collection	Read
message.formstring	Value of form string in the message	Proxy request	String	Read

Variable	Description	Scope	Type	Permission
message.header. {header_name}	Gets or sets the value of the specified HTTP header in the message	Proxy request	String	Read/Write
message.reason.phrase	Scope begins: Target response Type: String Permission: Read ReasonPhrase of the response message from target	Proxy request	String	Read
message.status.code	HTTP status code of the response message from target	Target response	Integer	Read
message.header. {header_name}.values	All values of the specified HTTP header name in the message	Proxy request	Collection	Read
message.header. {header_name}.values.count	Count of the values of the specified HTTP header name in the message	Proxy request	Integer	Read
message.headers.count	Count of all HTTP headers in the message	Proxy request	Integer	Read
message.headers.names	Value of all HTTP headers in the message	Proxy request	Collection	Read
messagelogging.{policy_name}.failed	Failure flag for the referenced Message logging policy	Proxy request	Boolean	Read
messagelogging.failed	Failure flag for Message logging policy	N/A	N/A	N/A
message.queryparam. {query_param_name}.values	Value of the specified query parameter in the message	Proxy request	Integer	Read
message.transport.message	Message of type <code>TransportMessage</code> which is a contextual object	Proxy request	<code>TransportMessage</code>	Read

## 1.5.1.4.2.9 Response Message Variables

Response message variables are used in policies to access message components like the header, the query parameters, form parameters, the source IP address, the HTTP message body.

API proxy applies the received response to a series of policies, depending on the request condition, API proxy can either modify or transform the request. Based on the content of the response variable, policies can either transform or reject the request. Supported response variables are listed in the Response Message Variables table.

Response Message Variables

Variable	Description	Scope	Type	Permission
client.sent.end.time	<p>The time, expressed in string form, at which the proxy finished sending the response from the ProxyEndpoint to the client. For example: Wed, 21 Aug 2013 19:16:47 UTC</p> <p>This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.</p>	Proxy response	String	Read
client.sent.end.timestamp	<p>The timestamp value specifying when the proxy finished sending the response to the client from the ProxyEndpoint. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.</p>	Proxy response	Long	Read

Variable	Description	Scope	Type	Permission
client.sent.start.time	The time, expressed in string form, at which the proxy began sending the response from the ProxyEndpoint to the client. For example: Wed, 21 Aug 2013 19:16:47 UTC  This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.	Proxy response	String	Read
client.sent.start.timestamp	The timestamp value specifying when the proxy began sending the response to the client from the ProxyEndpoint. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Proxy response	Long	Read
message.reason.phrase	ReasonPhrase of the response message from target	Target response	String	Read
message.status.code	HTTP status code of the response message from target	Target response	Integer	Read
response	Complete response message returned by target	Target response	Message	Read/Write
response.content	Payload content of the response message returned by the target	Target response	String	Read/Write
response.formparam.{formparam_name}	The value of a form parameter in the response	Target response	String	Read/Write
response.formparam.{formparam_name}.values.count	Count of all the values of the specified form parameter in response	Target response	Integer	Read
response.formparams.count	Count of all form parameters in the response	Target response	Integer	Read

Variable	Description	Scope	Type	Permission
response.formparams.names	The names of all the form parameters in the response	Target response	Collection	Read
response.header.{header_name}	Gets or sets the value of a specified HTTP header in the response. If the header has multiple values (such as a CSV list), a GET returns the first value only.	Target response	String	Read/Write
response.header.{header_name}.values	All the values of a specified HTTP header in response	Target response	Collection	Read
response.header.{header_name}.values.count	Count of all the values of the specified HTTP header in response	Target response	Integer	Read
response.headers.count	Count of all the headers in the response	Target response	Integer	Read
response.headers.names	The names of all the headers in the response	Target response	Collection	Read
response.reason.phrase	The response reason phrase for a particular request	Target response	String	Read/Write
response.status.code	The response code returned for a request. You can use this variable to override the response status code, which is stored in message.status.code.	Target response	Integer	Read/Write
response.transport.message	Response of type TransportMessage which is a contextual object	Target response	String	Read
target.country	Country/region of the TLS/SSL certificate presented by the target server	Target response	String	Read
target.email.address	E-mail address of the TLS/SSL certificate presented by the target server	Target response	String	Read
target.organization	Organization of the TLS/SSL certificate presented by the target server	Target response	String	Read

Variable	Description	Scope	Type	Permission
target.organization.unit	Organization unit of the TLS/SSL certificate presented by the target server	Target response	String	Read
target.locality	Locality (city) of the TLS/SSL certificate presented by the target server	Target response	String	Read
target.received.end.time	This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.	Target response	String	Read
target.received.end.timestamp	The timestamp value specifying when the TargetEndpoint finished receiving the response from the target. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Target response	Long	Read
target.received.start.time	The time, expressed in string form, at which the TargetEndpoint finished receiving the response from the target. For example: Wed, 21 Aug 2013 19:16:47 UTC  This time value is the string representation of the corresponding 32-bit timestamp quantity. For example, 'Wed, 21 Aug 2013 19:16:47 UTC' corresponds to the timestamp value of 1377112607413.	Target response	String	Read
target.received.start.timestamp	The timestamp value specifying when the TargetEndpoint started receiving the response from the target. This value is a 64-bit (long) integer containing the number of milliseconds elapsed since midnight, on January 1, 1970 UTC.	Target response	Long	Read

Variable	Description	Scope	Type	Permission
target.state	State of the TLS/SSL certificate presented by the target server	Target response	String	Read
target.host	The domain name of the target service returning the response to the API proxy.	Target response	String	Read
target.ip	The IP address of the target service returning the response to the API proxy.	Target response	String	Read
target.port	The port number of the target service returning the response to the API proxy.	Target response	Integer	Read
target.scheme	Returns http or https depending on the request message	Target response	String	Read/Write

### 1.5.1.4.2.10 Path Variables

Supported path variables are listed in the Path Variables table.

Path Variables

Variable	Path	Description
Request variables	request.uri	The HTTP request path, which includes a path and a query string separated by a question mark ( ? )
	request.path	The HTTP request path without the query string
	request.querystring	The portion of the HTTP request path after the question mark ( ? )
Application variables	application.basepath	The deployment base path (specified during API deployment)
Proxy variables	proxy.basepath	The base path as configured in the proxy XML file.
	proxy.pathsuffix	The portion of the request path after the proxy basepath, which is determined by any conditional flow URIs
Target variables	request.url	The complete URL of the final request made.
	target.basepath	The path (without host or port) in the URL configured in the target XML file or the dynamic target URL (if target.url is set during the message flow)

Variable	Path	Description
	target.url	The URL configured in the target XML file or the dynamic target URL (if target.url is set during the message flow). Returns null if called out of scope or otherwise unset.

## 1.5.1.4.2.11 Multi-value HTTP Headers in an API Proxy

Access and extract multiple values in an HTTP header.

An HTTP header is a name value pair that allows a client application or a backend system to pass additional information about requests and responses. Following are a few examples of simple http headers:

- `Retry-After: Wed, 05 May 2020 23:59:59 GMT`  
The `Retry-After` request header passes the information about how long the service is unavailable to the requesting client.
- `Server: APIM/3.0`  
The `Server` header passes the information about the software used by the origin server to handle the request.

An HTTP header can have multiple values depending on <header definition list link>. A multi-valued HTTP header has comma-separated values. Following are a few examples of headers that contain multiple values:

- `Content-Language: en, de, fr`
- `Content-Type: application/json, application/xml, text/html`
- `Cache-Control: no-transform, no-store, proxy-revalidate`
- `Accept: text/*, text/html, text/html;level=1, */*`

API Management allows developers to access HTTP headers in policies or in conditional flows using flow variables. Following are a list of variables in API Management that you can use to access a specific HTTP request or response header:

- `request.header.{header_name}`
- `request.header.{header_name}.values`
- `request.header.{header_name}.values.count`
- `request.headers.count`
- `response.header.{header_name}`
- `response.header.{header_name}.values`
- `response.header.{header_name}.values.count`
- `response.headers.count`

Following is a sample AssignMessage policy that shows how to read the value of a request header and store it in a variable:

### Sample Code

```
<AssignMessage async="false" continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
 <AssignVariable>
```



```
<Name>var</Name>
 <Ref>request.header.temp</Ref>
</AssignVariable
</AssignMessage>
```

## Accessing Multiple Values in an HTTP Header

Accessing the values of HTTP headers in API Management policies wherein only the first value of the header is returned is incorrect. The approach of extracting only one value of an HTTP header can lead to issues if the specific header has more than one value. Following are a few examples of multi-value header access:

- **Example 1:** Read a multi-valued header using javascript code  
Consider that the `Accept` header has multiple values as shown below:  
`Accept: application/xml, text/html, application/xhtml+xml`  
Following is sample JavaScript code that reads the value of `Accept` header:

### Sample Code

```
var valuesofacceptheader = context.getVariable("request.header.Accept");
```

The above sample code returns only the first value of the `Accept` header, which is `application/xml`.

- **Example 2:** Read a multi-valued header in Assign Message Policy  
Consider that the `Access-Control-Allow-Headers` header has multiple values as shown below:  
`Access-Control-Allow-Headers: content-type, authorization`  
Following is the sample Assign message policy payload that sets the value of `Access-Control-Allow-Headers` header:

### Sample Code

```
<AssignMessage async="false" continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
 <Headers>
 <Header name="Access-Control-Allow-Headers">{request.header.Access-
Control-Allow-Headers}</Header>
 </Headers>
 <IgnoreUnresolvedVariables>>false</IgnoreUnresolvedVariables>
 <AssignTo createNew="true" transport="http" type="response"/>
</AssignMessage>
```

The above sample code sets the `Access-Control-Allow-Headers` header with only the first value, which is `content-type`.

In both the examples above, only the first value of the multi-valued headers are returned. Later, if any other policy in an API proxy tries to use these values to perform some function, then it could lead to errors.

## Extracting Multiple Values in an HTTP Header

To extract multiple values in an HTTP header, you can use the relevant built-in flow variables such as `request.header.{header_name}.values.count`, `request.header.{header_name}.values`, `response.header.{header_name}.values.count`, and `response.header.{header_name}.values`.

You can then iterate to retrieve all the values from a specific header using a sample JavaScript code as shown below:

### Sample Code

```
for (var i = 1; i <=context.getVariable('request.header.Content-Type.values.count'); i++)
{
 print(context.getVariable('request.header.Content-Type' + i));
}
```

### 1.5.1.4.3 CSRF Token Handling in API Management

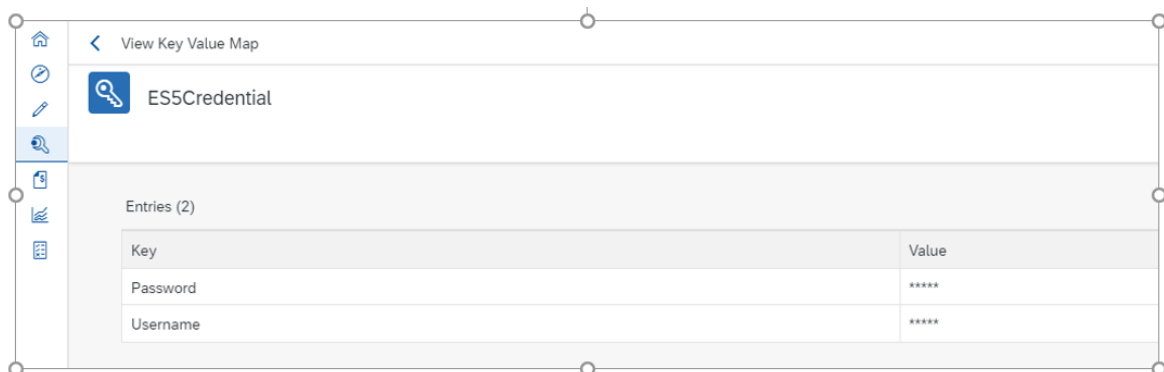
While exposing any back-end services, like SAP GW OData service via API Management, you have to enable server to server authentication between the back-end and API Management system.

You enable the server to server authentication, so that the API consumers don't have to know the back-end system credentials, instead they can use the policy based authentication. Whenever you make a POST, PUT, and a DELETE HTTP verb, it requires a CSRF token validation from the back-end. In this section, we've described how to automate the CSRF token validation from the API Management layer.

The policies required to enable back-end authentication from API Management.

## Enabling Backend Authentication from API Management

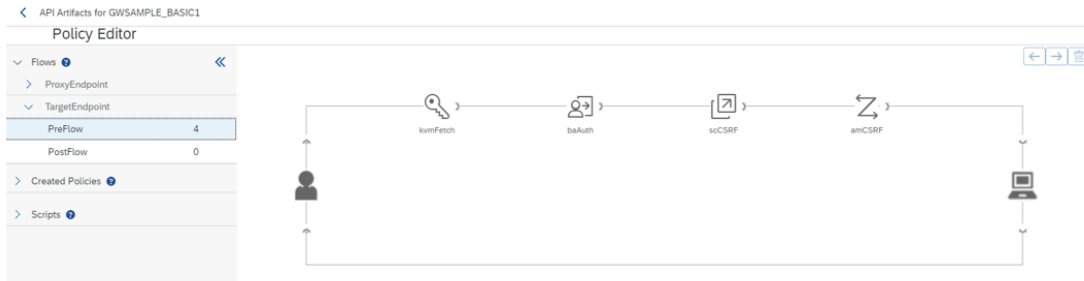
1. Create a Key value map for storing credential of back-end system.



2. Implement the following 4 policies in the target endpoint:
  - **kvmFetch (KeyValueMapOperations)**: Use this policy to retrieve the username and password from the key value map and assign them to the private variables.

## Note

Condition strings aren't required for this policy.



```
<KeyValueMapOperations mapIdentifier="ES5Credential"
continueOnError="false" enabled="true" xmlns="http://www.sap.com/apimgmt">
 <Get assignTo="private.BasicAuthUsername" index='1'>
 <Key><Parameter>Username</Parameter></Key>
 </Get>
 <Get assignTo="private.BasicAuthPassword" index='1'>
 <Key><Parameter>Password</Parameter></Key>
 </Get>
 <Scope>environment</Scope>
</KeyValueMapOperations>
```

- **baAuth (BasicAuthentication)**: Use this policy to encode the username and password in Base64 and assign that to the authorization header.

## Note

Condition strings aren't required for this policy.

```
<BasicAuthentication continueOnError='false' enabled='true' xmlns='http://
www.sap.com/apimgmt'>
 <Operation>Encode</Operation>
 <IgnoreUnresolvedVariables>>false</IgnoreUnresolvedVariables>
 <User ref='private.BasicAuthUsername'></User>
 <Password ref='private.BasicAuthPassword'></Password>
 <AssignTo createNew="true">request.header.Authorization</AssignTo>
</BasicAuthentication>
```

- **scCSRF (ServiceCallout)**: Use this policy to fetch the CSRF token from the back-end call. Use the following condition string:

```
(request.verb = "POST" OR request.verb = "PUT" OR request.verb = "DELETE")
```

Here we've used an API provider named ES5 for the back-end connection in API Management.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceCallout async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <Request>
 <Set>
 <Headers>
 <Header name="x-csrf-token">fetch</Header>
 <Header name="Authorization">{request.header.Authorization}</Header>
 </Headers>
 <Verb>GET</Verb>
 </Set>
 <IgnoreUnresolvedVariables>>false</IgnoreUnresolvedVariables>
```

```

</Request>
 <Response>callOutResponse</Response>
 <Timeout>30000</Timeout>
 <HTTPTargetConnection>
 <APIProvider>ES5</APIProvider>
 <Path>sap/opu/odata/IWFND/CATALOGSERVICE/ServiceCollection</Path>
 </HTTPTargetConnection>
</ServiceCallout>

```

- **amCSRF (AssignMessage):** Use this policy to assign the CSRF token and the cookies to the request message. Use the following condition string:

```
(request.verb = "POST" OR request.verb = "PUT" OR request.verb = "DELETE")
```

```

<!-- This policy can be used to create or modify the standard HTTP request
and response messages -->
<AssignMessage async="false" continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
 <!-- Sets a new value to the existing parameter -->
 <Set>
 <Headers>
 <Header name="x-csrf-token">{callOutResponse.header.x-csrf-token}</
Header>
 <Header name="Cookie">{callOutResponse.header.Set-Cookie.1};
{callOutResponse.header.Set-Cookie.2};{callOutResponse.header.Set-
Cookie.3}</Header>
 </Headers>
 </Set>
 <IgnoreUnresolvedVariables>>false</IgnoreUnresolvedVariables>
 <AssignTo createNew="false" type="request">request</AssignTo>
</AssignMessage>

```

### Note

The number of cookies is back-end specific. Therefore, set the number of cookie attributes in the cookie header accordingly.

## Conclusion

Alternatively, you can implement the Service Callout policy for on-premise APIs as shown below:

This implementation uses a local proxy call in the Service Callout policy.

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceCallout async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <Request>
 <Set>
 <Headers>
 <Header name="x-csrf-token">fetch</Header>
 <Header name="Authorization">{request.header.Authorization}</Header>
 </Headers>
 <Verb>GET</Verb>
 </Set>
 <IgnoreUnresolvedVariables>>false</IgnoreUnresolvedVariables>
 </Request>
 <Response>callOutResponse</Response>
 <Timeout>30000</Timeout>
<LocalTargetConnection>

```

```

 <Path><API Base Path></Path>
 </LocalTargetConnection>
</ServiceCallout>

```

## 1.5.1.5 File Resource

File resource is a script or code snippet that can be attached to flows using policies.

An API proxy container supports definition of a number of Java, Python, or XSL scripts. These scripts can be executed in the context of either a java script, python script, or XSL transformation policy. Once a script is defined, it can be applied as a either a java script, python script, or XSL transformation policy in different flows.

## 1.5.1.6 API Proxy Structure

During an import or export of an API proxy, the API proxy follows a specific predefined structure.

The structure of an API proxy is as follows:

API Proxy Structure

Folder Name	Path	Contents
APIProxy	\API Proxy	Root folder that contains the APIProxyEndPoint, APITargetEndPoint, APIResource, Documentation, FileResource, and Policy information.
APIProxyEndPoint	\API Proxy\APIProxyEndPoint	This folder has a <b>&lt;Proxy Endpoint name&gt;.xml</b> file that contains information about Proxy Endpoint, resources, documentation, and the policy assignments on the proxy endpoint stream.
APIResource	\API Proxy\APIResource\	<b>Contains one file for every resource associated with the APIProxy. Each &lt;API Resource name&gt;.xml</b> file contains the API resource details such as resource name, title, documentation, and so on.

Folder Name	Path	Contents
Documentation	\API Proxy\Documentation	Contains one documentation file for every resource. Each document follows the naming convention <b>&lt;APIResource name&gt;_&lt;language&gt;.html</b> . For example, if the API Resource name is <i>PurchaseOrder</i> and the supported language is <i>English</i> , then the document file name is <i>PurchaseOrder_en.html</i> . The file provides the documentation content relevant to the associated resource.
FileResource	\API Proxy\FileResource	Lists all the scripts attached to the policy. Only Java, Python, and XSL Scripts are supported. Follow the below naming convention: <ul style="list-style-type: none"> <li>• Java Script: <b>&lt;JavaScript name&gt;.js</b></li> <li>• Python script: <b>&lt;PythonScript name&gt;.py</b></li> <li>• XSL script: <b>&lt;XSLScript name&gt;.xsl</b></li> </ul>
Policy	\API Proxy\Policy	Contains a list of all policies attached to the API Proxy. Each policy is available as a separate file with the naming convention <b>&lt;Policy name&gt;.xml</b> . The folder should also contain a <i>defaultRaiseFaultPolicy.xml</i> .
<b>&lt;APIProxyName&gt;.xml</b>	\API Proxy\ <b>&lt;APIProxyName&gt;.xml</b>	This file contains the header information of the proxy endpoint, target endpoint, policies, and file resources.

### Note

- When you import an API proxy, ensure that the API proxy name in the **<APIProxy name>.xml** file and the value of the *base\_path* field (inside the *APIProxyEndpoint* file) is unique.
- If the API proxy does not contain any API resources, then the *APIResources*, *Documentation*, *FileResource*, and *Policy* folders are not required in the .zip file during import of API proxy.

## Route

A proxy endpoint can connect to one or more Target Endpoints. A route connects the proxy endpoint to the Target Endpoint. It determines which Target Endpoint to invoke based on the proxy endpoint configuration.

### Note

You can also have an empty route, which means you do not define a Target Endpoint. You can define such routes when you do not want to forward the request message to any Target Endpoint. For example, flows that generate OAuth token.

## Route Rule

All requests are forwarded to the respective Target Endpoints by implementing certain rules on the route. The Route Rule is basically a conditional statement that determines the Target Endpoint. When more than one Target Endpoint is available, the Route Rule evaluates the condition and, if true, the request is forwarded to the named target endpoint.

For more information on how to define multiple target endpoints using Route Rule, see [Enable Dynamic Routing \[page 596\]](#)

## Fault Rule

A lot of error conditions can arise when API proxies are servicing requests from applications. For example, you may encounter network issues when communicating with backend services, applications may present expired credentials, request messages may be incorrectly formatted, and so on. In such cases, it is important to handle these errors in a customized manner. When an API proxy encounters an error, the default behavior is to exit from the normal processing pipeline and to enter an error Flow. This error Flow bypasses any remaining processing Steps and Policies. As a result the raw error message or codes are returned to the requesting application. It is important to modify this behavior and improve usability. The API Platform enables you to customize exception handling by defining Fault Rules. Fault Rules can be attached to proxy endpoints, target endpoints, and Route rules. A Fault Rule is an XML configuration element that specifies:

- A condition that classifies a fault based on the predefined category, subcategory, or name of the fault
- One or more Policies that define the behavior of the FaultRule

### 1.5.1.7 Endpoint Property Reference

This topic describes transport properties that can be set in TargetEndpoint and ProxyEndpoint configurations to control messaging and connection behavior.

#### [Proxy Endpoint Properties \[page 471\]](#)

ProxyEndpoint HTTPTargetConnection elements define a set of HTTP transport properties. These properties can be used to set transport-level configurations.

#### [Target Endpoint Properties \[page 473\]](#)

HTTP transport properties configured in the HTTPTargetConnection element in TargetEndpoint configurations defines a set of properties to set transport level configurations.

### 1.5.1.7.1 Proxy Endpoint Properties

ProxyEndpoint HTTPTargetConnection elements define a set of HTTP transport properties. These properties can be used to set transport-level configurations.

You can export the proxy endpoint as a zip file and add the ProxyEndpoint properties to the the APIProxyEndPoint subfolder under the APIProxy parent folder. For more information, see [Export an API Definition \[page 533\]](#).

Properties are set on ProxyEndpoint HTTPProxyConnection elements as follows:

### Sample Code

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ProxyEndPoint default="true">
 <name>default</name>
 <base_path>/sdf</base_path>
 <properties>
 <property>
 <name>temp</name>
 <value>v1</value>
 </property>
 </properties>
 <routeRules>
 <routeRule>
 <name>default</name>
 <targetEndPointName>default</targetEndPointName>
 <sequence>1</sequence>
 <faultRules/>
 </routeRule>
 </routeRules>
 <faultRules/>
 <preFlow>
 <name>PreFlow</name>
 </preFlow>
 <postFlow>
 <name>PostFlow</name>
 </postFlow>
 <conditionalFlows/>
</ProxyEndPoint>
```

### Note

Alternatively, you can navigate to the **Design > APIs** tab, choose the API proxy that you deployed from the APIs list, and select the *Proxy EndPoint* tab. In the *Proxy Endpoint Properties* section, choose *Add* and enter the *Property Name* and the *Value* as defined in the **ProxyEndpoint Transport Property Specification** table.

## ProxyEndpoint Transport Property Specification

ProxyEndpoint Transport Property Specification

Property Value	Default Value	Description
X-Forwarded-For	false	On setting it to true, the virtual host's IP address is added to the outbound request as the value of the HTTP X-Forwarded-For header.
request.streaming.enabled	false	<p>Default value (false): HTTP request payloads are read into a buffer, and policies operating on the payload work as expected.</p> <p>true: HTTP request payloads are not read into a buffer, they are streamed to the TargetEndpoint request flow. And, policies operating on the payload in the ProxyEndpoint request flow are bypassed.</p> <p>For more information about enabling streaming, see <a href="#">Enable Streaming of Requests and Responses in an API Proxy [page 594]</a></p>



Property Value	Default Value	Description
response.streaming.enabled	false	<p>Default value (false): HTTP response payloads are read into a buffer, and policies operating on the payload work as expected.</p> <p>true: HTTP response payloads are not read into a buffer, they are streamed to the TargetEndpoint request flow. And, policies operating on the payload in the ProxyEndpoint response flow are bypassed.</p> <p>For more information about enabling streaming, see <a href="#">Enable Streaming of Requests and Responses in an API Proxy [page 594]</a></p>
compression.algorithm	N/A	<p>Supported values:</p> <ul style="list-style-type: none"> <li>gzip: always send message using gzip compression</li> <li>deflate: always send message using deflate compression</li> <li>none: always send message without any compression</li> </ul> <p>For example, when a client submits a request that uses <code>gzip</code> compression, API Management forwards the request to target using <code>gzip</code> compression. You can configure compression algorithms to be explicitly applied by setting this property on the TargetEndpoint or ProxyEndpoint.</p>
api.timeout	N/A	Configure the timeout for individual API proxies.

Parent topic: [Endpoint Property Reference \[page 471\]](#)

## Related Information

[Target Endpoint Properties \[page 473\]](#)

### 1.5.1.7.2 Target Endpoint Properties

HTTP transport properties configured in the HTTPTargetConnection element in TargetEndpoint configurations defines a set of properties to set transport level configurations.

You can export the proxy endpoint as a zip file and add the TargetEndpoint properties to the APITargetEndPoint subfolder under the APIProxy parent folder. For more information, see [Export an API Definition \[page 533\]](#).

Configure the properties on TargetEndpoint HTTPTargetConnection elements:

#### Sample Code

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<TargetEndPoint>
 <name>default</name>
 <url>http://www.abc.com</url>
 <provider_id>NONE</provider_id>
 <isDefault>true</isDefault>
 <properties>
```

```

 <property>
 <name>temp</name>
 <value>value</value>
 </property>
 <property>
 <name>test</name>
 <value>v2</value>
 </property>
 </properties>
 <faultRules/>
 <preFlow>
 <name>PreFlow</name>
 </preFlow>
 <postFlow>
 <name>PostFlow</name>
 </postFlow>
 <conditionalFlows/>
</TargetEndPoint>

```

### Note

Alternatively, you can navigate to the **Design > APIs** tab, choose the API proxy that you deployed from the APIs list, and select the *Target EndPoint* tab. In the *Target Endpoint Properties* section, choose *Add* and enter the *Property Name* and the *Value* as defined in the **TargetEndpoint Transport Property Specification** table.

## TargetEndpoint Transport Property Specification

TargetEndpoint Transport Property Specification

Property Name	Default Value	Description
keepalive.time-out.millis	60000 milliseconds	Connection idle timeout for the target connection in the connection pool. If the connection in the pool is idle beyond the specified limit, then the connection is closed.
connect.time-out.millis	3000 milliseconds	Target connection timeout. returns an HTTP 503 status code if a connection timeout occurs.

Property Name	Default Value	Description
io.timeout.millis	55000 milliseconds	<p>If there's no data to read for the specified number of milliseconds, or if the socket is not ready to write data for specified number of milliseconds, then the transaction is treated as a timeout.</p> <ul style="list-style-type: none"> <li>If a timeout happens while writing the HTTP request, 408, Request Timeout is returned.</li> <li>If a timeout happens while reading the HTTP response, 504, Gateway Timeout is returned.</li> </ul>
<div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p><b>Note</b></p> <p>In general, the default timeout value is 55 seconds for APIs. The APIs created based on the on-premise backend, the value is 54 seconds.</p> <p>For the API proxies where the io.timeout.millis value isn't set explicitly in the target endpoint, the default timeout value of 55 seconds or 54 seconds (based on the type of the backend system) is considered in the runtime.</p> <p>For the API proxies where the io.timeout.millis value is explicitly set in the target endpoint, the same value is considered in the runtime, provided it's less than the default 55 seconds/54 seconds. Please note that the older version of API proxies created based on the on-premise backend system before July 2022 will require a redeployment for the timeout propagation to come into effect.</p> <p>Also, note that the timeout explained above is not precise, the actual timeout could occur slightly earlier than the timeout value.</p> </div>		
supports.http10	true	If true and the client sends a 1.0 request, the target is also sent a 1.0 request. Otherwise 1.1 request is sent to target.
supports.http11	true	If true and the client sends a 1.1 request, the target is also sent a 1.1 request, otherwise 1.0 request is sent to target.
success.codes	N/A	<p>By default, HTTP code 4XX or 5XX are treated as errors, HTTP code 1XX, 2XX, 3XX as success. This property enables explicit definition of success codes, for example, 2XX, 1XX, 505 treats any 100, 200 and 505 HTTP response codes as success.</p> <p>Setting this property overwrites the default values. Therefore, if you want to add HTTP code 400 to the list of default success codes, set this property as:</p> <pre>&lt;Property name="success.codes"&gt;1xx,2xx,3xx,400&lt;/Property&gt;</pre> <p>If you want only HTTP code 400 to be treated as a success code, set the property as:</p> <pre>&lt;Property name="success.codes"&gt;400&lt;/Property&gt;</pre> <p>By setting HTTP code 400 as the only success code, the codes 1xx, 2xx, and 3xx are treated as failures.</p>

Property Name	Default Value	Description
compression.algorithm	N/A	<p>Supported values:</p> <ul style="list-style-type: none"> <li>gzip: always send message using gzip compression</li> <li>deflate: always send message using deflate compression</li> <li>none: always send message without any compression</li> </ul> <p>For example, when a client submits a request that uses <code>gzip</code> compression, API Management forwards the request to target using <code>gzip</code> compression. You can configure compression algorithms to be explicitly applied by setting this property on the <code>TargetEndpoint</code> or <code>ProxyEndpoint</code>.</p>
enable.method.override	false	<p>For the specified HTTP method, sets an X-HTTP-Method-Override header on the outbound request to the target service. For example, <code>&lt;Property&gt;&lt;name&gt;="GET.override.method"&lt;/name&gt;&lt;value&gt;POST&lt;/value&gt;&lt;/Property&gt;</code>.</p>
request.streaming.enabled	false	<p>Default value (false): HTTP request payloads are read into a buffer, and policies operating on the payload work as expected.</p> <p>True: HTTP request payloads are not read into a buffer, they are streamed to the target endpoint. And, policies that operate on the payload in the <code>TargetEndpoint</code> request flow are bypassed.</p> <p>For more information about enabling streaming, see <a href="#">Enable Streaming of Requests and Responses in an API Proxy [page 594]</a></p>
*.override.method	N/A	<p>For the specified HTTP method, sets an X-HTTP-Method-Override header on the outbound request. For example, <code>&lt;Property&gt;&lt;name&gt;="GET.override.method"&lt;/name&gt;&lt;value&gt;POST&lt;/value&gt;&lt;/Property&gt;</code>.</p>
response.streaming.enabled	false	<p>Default value (false): HTTP response payloads are read into a buffer, and policies that can operate on the payload work as expected.</p> <p>true: HTTP response payloads aren't read into a buffer; they're streamed as-is to the <code>ProxyEndpoint</code> response flow. In this case, any policies that operate on the payload in the <code>TargetEndpoint</code> response flow are bypassed.</p> <p>For more information about enabling streaming, see <a href="#">Enable Streaming of Requests and Responses in an API Proxy [page 594]</a></p>
request.retain.headers.enabled	true	<p>By default, all HTTP headers on outbound messages are retained. On setting it to true, all HTTP headers present on the inbound request are set on the outbound request.</p>
request.retain.headers	N/A	<p>Set HTTP headers from the request on the outbound request to the target service. For example, to 'passthrough' the <code>User-Agent</code> header, set the value of <code>request.retain.headers</code> to <code>User-Agent</code>. Specify multiple HTTP headers as a comma-separated list, for example, <code>User-Agent,Referer,Accept-Language</code>. This property overrides <code>request.retain.headers.enabled</code>. If <code>request.retain.headers.enabled</code> is set to false, any headers specified in the <code>request.retain.headers</code> property are still set on the outbound message.</p>

Property Name	Default Value	Description
response.retain.headers.enabled	true	By default, all HTTP headers on outbound messages are retained. On setting it to true, all HTTP headers present on the inbound response from the target service are set on the outbound response before it's passed to the ProxyEndpoint.
response.retain.headers	N/A	Set HTTP headers from the response on the outbound response before it is passed to the ProxyEndpoint. For example, to passthrough the Expires header, set the value of response.retain.headers to Expires. Specify multiple HTTP headers as a comma-separated list, for example, Expires,Set-Cookie. This property overrides response.retain.headers.enabled. If response.retain.headers.enabled is set to false, any headers specified in the response.retain.headers property are still set on the outbound message.
retain.queryparams	N/A	Set query parameters on the outbound request. For example, to include the query parameter apikey from the request message, set retain.queryparams to apikey. Specify multiple query parameters as a comma-separated list, for example, apikey, environment. This property overrides retain.queryparams.enabled.
retain.queryparams.enabled	true	By default, all query parameters on outbound requests are retained. On setting it to true, all query parameters present on the inbound request are set on the outbound request to the target service.

Parent topic: [Endpoint Property Reference \[page 471\]](#)

## Related Information

[Proxy Endpoint Properties \[page 471\]](#)

## 1.5.2 Different Methods of Creating an API Proxy

An API proxy is the data object that contains all the functionality to be executed when an external user wants to access the backend service.

When you create an API, you can choose from the following options:

### *Import API*

If you have an API definition, you can reuse it by importing it into the . For more information, see [Import an API Definition \[page 534\]](#).

### *Create*

: Create an API proxy from scratch by defining the resources, documentation, and by attaching policies. For more information, see [Create an API Proxy \[page 478\]](#).

Model an API from a specification that contains the single target URL endpoint using the API Designer. For more information, see [Create an API Proxy Using the API Designer \[page 488\]](#).

### Note

- supports OData, REST, and SOAP services.
- An API proxy consists of a virtual host and a base path. The base path can be identical for multiple API proxies, provided API proxies have different virtual hosts. This means, for an API proxy, the combination of the virtual host and base path should be unique.  
The example below explains the same, where AP1 is proxy 1, AP2 is proxy 2, VH1 is Virtual Host 1, VH2 is the Virtual Host 2, and BP(A) is the base path.  
Example: AP1 = VH1+ BP(A), AP2 = VH2 + BP(A)

You can build a real-life API proxy that exposes a Web service from a backend system, and define policies to set rules on the API, for example, to enforce security or control API traffic. To know more, see [Policies \[page 205\]](#) and [Create a Policy \[page 583\]](#).

You can group API proxies together into units that represent all the services needed to perform some larger business under a data object known as a product, or create a product when you want to expose one or more APIs to the application developer. For more information, see [Create a Product \[page 653\]](#).

You can also view the applications you've subscribed to. To know more, see [View Applications \[page 660\]](#).

Parent topic: [Build API Proxies \[page 195\]](#)

## Related Information

[Key Components of an API \[page 196\]](#)

[Additional Configurations \[page 537\]](#)

### 1.5.2.1 Create an API Proxy

This topic describes the steps to create an API proxy from the .

## Prerequisites

The role collection *APIPortal.Administrator* should be assigned to you.

To build APIs, you must do the following:

- Make sure that you have the REST, OData, or SOAP URL of the service that you want to expose as an API.
- Browse for a service on a specific API provider. To do so, you must configure the required API provider on the [Configure](#) tab.
- You've created an instance against ORG/USER secret for creating an API Proxy from an Open Connector type API Provider.

## Context

Instead of consuming services directly, application developers can access API proxies exposed via API Management. You do so, by creating an API proxy that camouflages the service you want to expose. The API maps a publicly available HTTP endpoint to back-end services. Creating this API proxy lets API Management handle the security and authorizations required to protect, analyze, and monitor your services.

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose ► [Configure](#) ► [APIs](#) ►.

A list of APIs appears in the catalog.

Alternatively, browse for a service on a specific API provider. To do so, you must configure the required API provider. You can view the number of calls made for an API in the current month. The data is visible for each API in the [Calls](#) column and also on the details screen of the individual API.

3. To expose a service as an API, choose ► [Create](#) ►.
4. Select the option based on your preference.

Select...	To...
<a href="#">API Provider</a>	<a href="#">Create an API Proxy by Referring to an API Provider System [page 483]</a>
<a href="#">API Proxy</a>	<a href="#">Create an API Proxy Based on an Existing API Proxy [page 485]</a>
<a href="#">URL</a>	<a href="#">Create an API Proxy by Providing a Direct Target Endpoint URL [page 486]</a>

5. Select the appropriate tabs. You can choose from the following, [Overview](#), [Proxy Endpoint](#), [Target Endpoint](#), [Resources](#), and [Revisions](#).

Tab	Details
<a href="#">Overview</a>	You can edit the following:

Tab	Details
	<ul style="list-style-type: none"> <li>• Name of the API</li> <li>• Host Alias</li> <li>• API Base Path</li> <li>• API State</li> <li>• API Description</li> </ul>
Proxy Endpoint	You can add the proxy endpoint and the route rules.
Target Endpoint	You can choose URL, API Provider, or API proxy, as the target endpoint as well as enter target endpoint rules.
Resources	<p>You can add resources.</p> <p>Resources refer to the individual endpoints or services that are exposed through APIs.</p>
Revision	Once an API is created, you can plan subsequent compatible changes to the API by creating a revision.

6. Add a resource to refer to individual endpoints or services.

**Note**

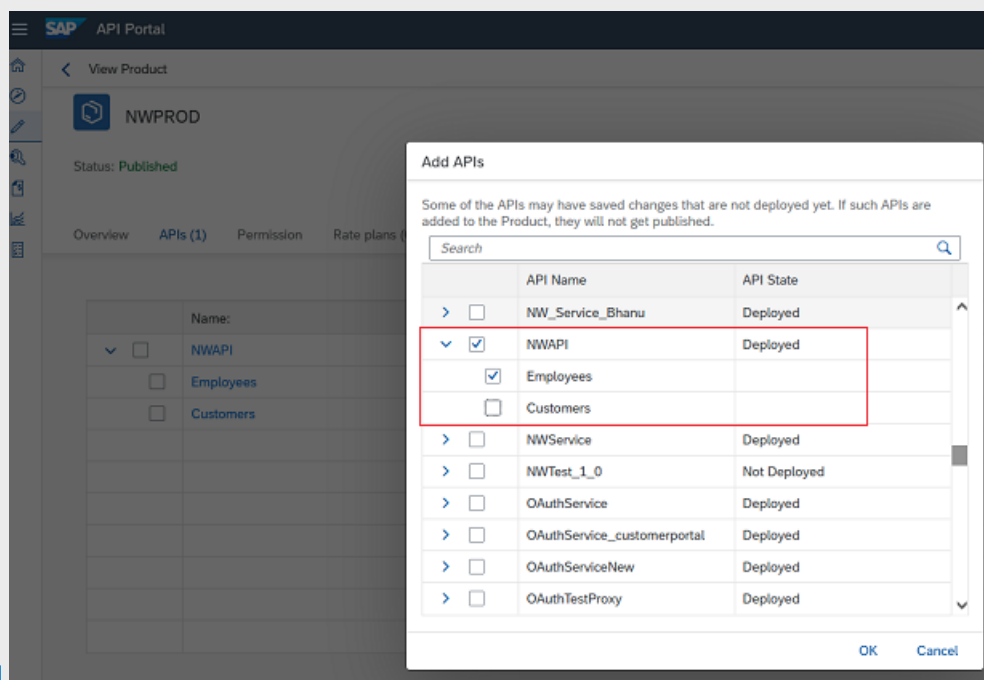
- For an OData service, all the associated artifacts appear on the different tab pages mentioned below. The [Resources](#) tab lists all the resources associated with the API. The API documentation with SAP documentation annotations, if selected, is also fetched from the metadata.
- For a SOAP- and REST-based service, the [Resources](#) tab appears. Add the resources manually.

**Note**

In case you want to restrict your users from accessing all the resources associated with the API Proxy, you need to create a new Product, add the required API to the Product, and select the



resources for which you want to provide access. For more information, see [Create a Product \[page 653\]](#).



653].

- For a *REST* service, add a resource as follows:
  1. Choose + (*Add*).
  2. In the popup, enter a title and path prefix for the resource.
  3. Select the methods that need to be supported for this resource.
  4. Add descriptions in the editor and choose *Add*.  
The added resource appears on the *Resources* tab.
  5. Choose *Add* to add more resources to the same API.
- For a SOAP service, add a resource as follows:
  1. Choose + (*Add*).
  2. Enter a title and specify the SOAP operation name in the path prefix.
  3. Use the editor to enter the relevant API documentation in the description field, and choose *Add*.  
The added resource appears on the *Resources* tab. By default, only the *POST* operation is selected.
  4. Add descriptions in the editor and choose *Add*.  
The added resource appears on the *Resources* tab.
  5. Choose *Add* to add more resources to the same API.
- For an OData service, select the methods that the application developer can perform:
  - *Get*: Read an entity.
  - *Post*: Create an entity.
  - *Put*: Update an entity.

### Note

API Management generates PUT operation as the default update operation. However, you can override the default update operation for API Proxy of type OData. For more information, see [Overriding the Default Update Operation for API Proxy of Type OData \[page 601\]](#).

- *Delete*: Delete an entity.

### Note

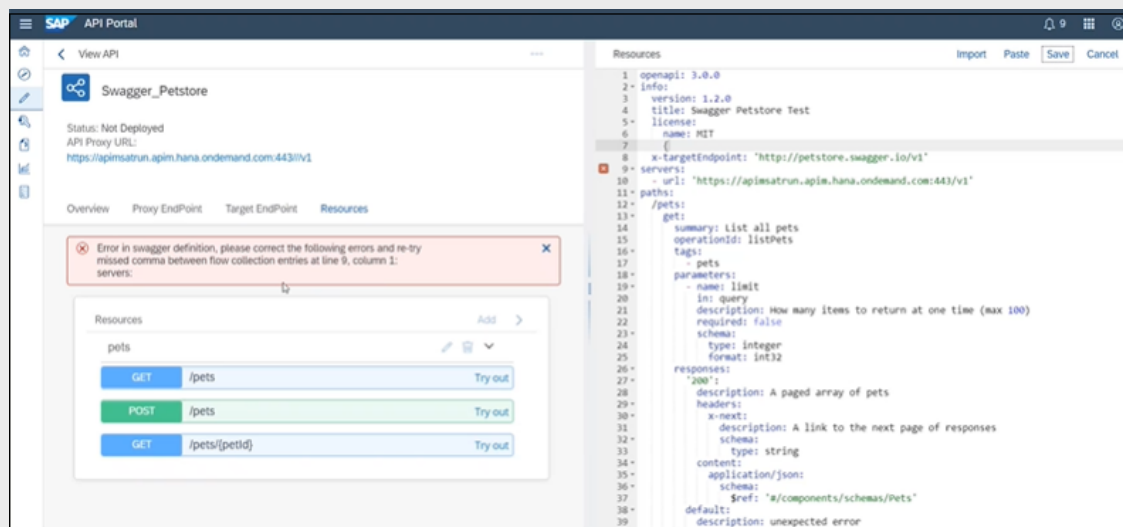
Only the supported methods for each resource appear on the UI. By default, only permitted methods are selected.

For a given resource, choose *Show/Hide* to view the list of properties and their associated API documentation. You can add descriptions for each resource in the editor.

### Note

For a given resource, choose *Open API Designer* and correct the errors in swagger definition, if any. The error message displayed on the screen helps in error detection and correction. Choose *Save* after making the necessary corrections in the swagger file.

See the example in the following screen:



7. To define policies on the API, go to the *Policies* tab. For more information about how to create a policy, see [Create a Policy \[page 583\]](#).
8. If you want to define multiple proxy endpoints, navigate to *Proxy EndPoint* tab.

In the *Proxy EndPoint Properties* section, choose *Add*. Enter the *Property Name* and the *Values*. For the Proxy Endpoint property specifications, see [Proxy EndPoint Properties \[page 471\]](#).

9. If you want to define multiple route rules, navigate to *Proxy EndPoint* tab. In the *Route Rules* section, choose *Add*.

### Note

When the API is created, the default route rule is set. It points to the default target endpoint and no rule is attached to it. Use the option *None* to ensure that no request is routed to any target endpoint. If there are multiple route rules, the rules are evaluated in sequence as displayed on the screen.

For more information on how to define multiple target endpoints using Route Rule, see [Enable Dynamic Routing \[page 596\]](#).

10. To define multiple target endpoints, navigate to the *Target EndPoint* tab and choose *Add*.

Enter the *Property Name* and the *Values*. For the Target Endpoint property specifications, see [Target Endpoint Properties \[page 473\]](#).

11. Once you've filled in all the required details of the API, you can select one of the following two actions for the API:

Action	Resulting API State	Future Action on API
<a href="#">Save</a>	<i>Not Deployed</i> : API is available only in the , and isn't available for product assignments.	<a href="#">Deploy</a> API is deployed and is ready for product assignments.
<a href="#">Deploy</a>	<i>Deployed</i> : Only deployed APIs can be selected for product publishing.	<a href="#">Undeploy</a> If any API is undeployed after being published, it's removed from the developer portal. When the API is deployed again, the product is updated. You can bring down an API without having to delete it from the product assignment. You can't undeploy an API if it's the only one associated with the product.

## Next Steps

Once you've created an API proxy, you can do the following:

## Related Information

[Create an API Proxy Using the API Designer \[page 488\]](#)

### 1.5.2.1.1 Create an API Proxy by Referring to an API Provider System

Create an API proxy by discovering or selecting one of the services available on the backend system configured by via the API provider.

**Prerequisite:** The role collection *APIPortal.Administrator* should be assigned to you.

## Browse for an OData Service for an API Provider

If you want to browse for an OData service for a provider that you've already configured, proceed as follows:

1. Log on to the .
2. Choose ► [Configure](#) ► [APIs](#) ▾ from the left navigation pane.
3. To expose a service as an API, choose [Create](#).
4. Select the [API Provider](#) radio button.
5. Select the required open connector type provider from the [API Provider](#) dropdown list.  
The dropdown list contains the providers that you're connected to. If the provider you need isn't listed here, add it on the [Configure](#) tab.
6. To view the list of OData services available in the provider, choose [Discover](#) and select the required service.  
If you deselect the [Link API Provider](#) checkbox, the API proxy is no longer linked to any API provider. It now acts just like a URL-based API. However, since the API created is originally of type OData, you can't add or delete resources to it.  
Also, since such APIs aren't linked to any API provider, you can't use the [Synchronize](#) option to update the API with the latest version that might be available in the backend.
7. Select a virtual host alias from the [Host Alias](#) dropdown.  
The details of the API name, description, API base path, and service type are automatically populated.
8. Optionally, enter a [Version](#) for your API proxy.  
When you choose to version your API proxy, its name is appended with the version, and its basepath are prepended with the version. For example, if the version you enter is v1, the name is Name\_v1, and the basepath is /v1/SalesOrder. For more information, see [API Versioning \[page 573\]](#).
9. Choose [Create](#).
10. If you want to add SAP documentation annotations to the API documentation, choose [Yes](#) for [Documentation](#).

#### Note

This field is only displayed if you're fetching services from SAP Gateway Systems. For more information about SAP documentation annotations, see [Extended Support of Long Texts in the Metadata](#).

11. Choose [Create](#).
12. Complete the remaining steps by referring to the [Create an API Proxy \[page 478\]](#) topic.

## Browse for an Open Connector Instance for an API Provider

If you want to create a proxy for an Open Connector instance for a provider that you've already configured, proceed as follows:

1. Log on to the .
2. Choose ► [Configure](#) ► [APIs](#) ▾ from the left navigation pane.
3. To expose a service as an API, choose [Create](#).
4. Select the [API Provider](#) radio button.
5. Select the required open connector type provider from the [API Provider](#) dropdown list.  
The dropdown list contains the providers that you're connected to. If the provider you need isn't listed here, add it on the [Configure](#) tab.
6. To view the list of OData services available in the provider, choose [Discover](#) and select the required service.  
If you deselect the [Link API Provider](#) checkbox, the API proxy is no longer linked to any API provider. It now acts just like a URL-based API. However, since the API created is originally of type OData, you can't add or delete resources to it.

Also, since such APIs aren't linked to any API provider, you can't use the [Synchronize](#) option to update the API with the latest version that might be available in the backend.

7. Select a virtual host alias from the [Host Alias](#) dropdown.  
The details of the API name, description, API base path, and service type are automatically populated.

#### Note

Name and basepath shouldn't contain spaces.

8. Optionally, enter a [Version](#) for your API proxy.  
When you choose to version your API proxy, its name is appended with the version, and its basepath are prepended with the version. For example, if the version you enter is v1, the name is Name\_v1, and the basepath is /v1/SalesOrder. For more information, see [API Versioning \[page 573\]](#).
9. Choose [Create](#).  
On creating the proxy, an encrypted key value map is created with the following name `apim.oc.instance.token` and key name `default`. Also, an open connector policy is attached to the incoming POST flow request of the target endpoint of the APIProxy.

#### Note

- For an API Proxy, you can have only one open connector policy attached and the content of the open connector policy can't be modified.
- An API Proxy consists of a virtual host and a base path. The base path can be identical for multiple API proxies, provided the API proxies have different virtual hosts. This means, for an API Proxy, the combination of the virtual host and base path should be unique.  
The example below explains the same, where AP1 is proxy 1, AP2 is proxy 2, VH1 is Virtual Host 1, VH2 is the Virtual Host 2, and BP(A) is the base path.  
Example: AP1 = VH1+ BP(A)  
AP2 = VH2 + BP(A)
- On deleting the proxy, the encrypted key value map created above is also deleted. Further, while exporting the API, encrypted key value map created above isn't exported with the API.

10. Choose [Create](#).
11. Complete the remaining steps by referring to the [Create an API Proxy \[page 478\]](#) topic.

## 1.5.2.1.2 Create an API Proxy Based on an Existing API Proxy

You can create a new API proxy based on an existing one, while also making any necessary modifications to suit your specific requirements.

**Prerequisite:** The role collection [APIPortal.Administrator](#) should be assigned to you.

If you want to browse for an existing API proxy, proceed as follows: Create an API Proxy Based on an Existing API Proxy

1. Log on to the .
2. Choose [Configure > APIs](#) from the left navigation pane.
3. To expose a service as an API, choose [Create](#).

4. Select the [API Proxy](#) radio button.
5. To view the list of API proxies available, choose [Discover](#).
6. Select the required API proxy.
7. Enter a name and description for the API.  
The dropdown list contains the providers that you're connected to. If the provider you need isn't listed here, add it on the [Configure](#) tab.
8. Optionally, enter a [Version](#) for your API proxy.  
When you choose to version your API proxy, its name is appended with the version, and its basepath are prepended with the version. For example, if the version you enter is v1, the name is Name\_v1, and the basepath is /v1/SalesOrder. For more information, see
9. Select a virtual host alias from the [Host Alias](#) dropdown.
10. In the [API Base Path](#) field, provide a path prefix for the API.
11. Choose [Create](#).
12. Complete the remaining steps by referring to the [Create an API Proxy \[page 478\]](#) topic.

### 1.5.2.1.3 Create an API Proxy by Providing a Direct Target Endpoint URL

Create an API proxy of the type REST and SOAP using the HTTP endpoint URL.

#### Prerequisite

The role collection [APIPortal.Administrator](#) should be assigned to you.

#### Create an API Proxy of the type OData

1. Select the [URL](#) radio button.
2. Enter the OData service URL in the [URL](#) field. For example, http://<host>:<port>/SFlight.

#### Note

Ensure that the service URL you provide doesn't redirect to a different URL. That is, check if the service URL you're trying to access is temporarily or permanently moved to a different location. If it does so, then it's recommended that you provide the new location (redirected URL, if exists) of the service. For more information about how to handle URL redirection, see [Handling URL Redirects in an API Proxy Using Policies \[page 590\]](#).

3. Enter a name and description for the API.
4. Optionally, enter a version for your API Proxy.
5. When you choose to version your API Proxy, its name is appended with the version, and its basepath is prepended with the version. For example, if the version you enter is v1, the name is Name\_v1, and the basepath is /v1/SalesOrder. For more information, see [API Versioning \[page 573\]](#).

6. Select a virtual host alias from the *Host Alias* dropdown.
7. In the *API Base Path* field, provide a path prefix for the API. For example, v1/SFlight.
8. In the *Service Type* field, enter **OData**.
9. Choose *Create*.
10. Complete the remaining steps by referring to the [Create an API Proxy \[page 478\]](#) topic.

## Create an API Proxy of the type REST

1. Select the *URL* radio button.
2. Enter the REST service URL in the *URL* field. For example, http://<host>:<port>/SFlight.

### Note

Ensure that the service URL you provide doesn't redirect to a different URL. That is, check if the service URL you're trying to access is temporarily or permanently moved to a different location. If it does so, then it's recommended that you provide the new location (redirected URL, if exists) of the service. For more information about how to handle URL redirection, see [Handling URL Redirects in an API Proxy Using Policies \[page 590\]](#).

3. Enter a name and description for the API.
4. Optionally, enter a version for your API Proxy.
5. When you choose to version your API Proxy, its name is appended with the version, and its basepath is prepended with the version. For example, if the version you enter is v1, the name is Name\_v1, and the basepath is /v1/SalesOrder. For more information, see [API Versioning \[page 573\]](#).
6. Select a virtual host alias from the *Host Alias* dropdown.
7. In the *API Base Path* field, provide a path prefix for the API. For example, v1/SFlight.
8. In the *Service Type* field, enter **REST**.
9. Choose *Create*.
10. Complete the remaining steps by referring to the [Create an API Proxy \[page 478\]](#) topic.

## Create an API Proxy of the type SOAP

1. Select the *URL* radio button.
2. Enter the SOAP service URL in the *URL* field. For example, http://<host>:<port>/SFlight.

### Note

Ensure that the service URL you provide doesn't redirect to a different URL. That is, check if the service URL you're trying to access is temporarily or permanently moved to a different location. If it does so, then it's recommended that you provide the new location (redirected URL, if exists) of the service. For more information about how to handle URL redirection, see [Handling URL Redirects in an API Proxy Using Policies](#).

3. Enter a name and description for the API.
4. Optionally, enter a version for your API Proxy.

5. When you choose to version your API Proxy, its name is appended with the version, and its basepath is prepended with the version. For example, if the version you enter is v1, the name is Name\_v1, and the basepath is /v1/SalesOrder. For more information, see [API Versioning \[page 573\]](#).
6. Select a virtual host alias from the *Host Alias* dropdown.
7. In the *API Base Path* field, provide a path prefix for the API. For example, v1/SFlight.
8. In the *Service Type* field, enter **SOAP**.
9. Choose *Create*.
10. Complete the remaining steps by referring to the [Create an API Proxy \[page 478\]](#) topic.

## 1.5.2.2 Create an API Proxy Using the API Designer

Model APIs in the Open API format that is available on the .

### Prerequisites

The role collection *APIPortal.Administrator* should be assigned to you.

To build API proxies, you must have the REST, OData, or SOAP URL of the service that you want to expose as an API.

### Context

The API designer editor includes rich inbuilt capabilities that enable you to do the following:

- Import existing open APIs
- Download APIs
- Generate equivalent HTML output views
- Save APIs
- Validate open API syntax

The API designer supports OAS 2.0 and OAS 3.0 versions.

To know more about OAS 3.0 support in API Management, see [OpenAPI Specification 3.0 \[page 489\]](#).

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose ► *Configure* > *APIs* > .

A list of APIs appears in the catalog.



- To model an API in Open API format, choose [▶ Create in API Designer ▶](#).

#### Note

- You can now create your API in the embedded API designer
- To update an existing API, select the API and choose [Edit in API Designer](#).

- Enter the API-related information and choose [Save](#). For information on how to model the API, see [API Designer \[page 490\]](#).
- You can choose to version your the API, by selecting [▶ Save ▶ Save as Version ▶](#), and entering a new value in the [Version](#) field. To know more about versioning your API, see [API Versioning \[page 573\]](#)

When you choose to version your API proxy, it's name will be appended with the version, and it's basepath will be prepended with the version. For example, If the version you enter is v1, the name will be Name\_v1, and the basepath will be /v1/SalesOrder.

## Related Information

[Create an API Proxy \[page 478\]](#)

### 1.5.2.2.1 OpenAPI Specification 3.0

In addition to supporting OAS 2.0, Integration Suite now supports OpenAPI Specification (OAS) 3.0.

Integration Suite supports creation and import of API definitions in Open API Specification(OAS) 3.0. These OAS 3.0 API definitions can be created using API designer or existing ones can be imported using import functionality. For the created or imported API definition, API proxy is created on which user can apply policies to bring in capabilities.

To know more about importing APIs, see [Import an API Definition \[page 534\]](#).

To know more about creating APIs using the API designer, see [Create an API Proxy Using the API Designer \[page 488\]](#).

## Restrictions

- External references aren't supported with OAS 3.0 (or OAS 2.0) in .
- Images aren't supported with OAS 3.0 in API Management.
- Local and remote links aren't supported with OAS 3.0 .
- API proxy creation is not possible by discovering the OAS 3.0 APIs from SAP Business Accelerator Hub.

## 1.5.2.2.2 API Designer

You can create APIs in API designer.

Use the [OpenAPI Specification](#)  to create APIs in API designer.

API specification can be written in YAML or JSON. In this guide, we use both YAML and JSON examples.

The API designer has a preview pane. You can write API specification in YAML in the right-hand pane and preview the corresponding reference documentation in the left-hand pane.

The following image illustrates the process of API creation in API designer.

### Note

All the API code samples shown in this document refers to only API 2.0 specification.

### Note

The image contains links to more information. You can hover over each shape for a description and click on it for more information.

Define Metadata

Define Root URL

Add Attributes

Define Operations

Define Parameters

Define Responses

Add Definitions

Add Security Definitions

Add Tags

Add External Links

- [#unique\\_194/unique\\_194\\_Connect\\_42\\_subsection-im1 \[page 492\]](#)
- [#unique\\_194/unique\\_194\\_Connect\\_42\\_subsection-im2 \[page 494\]](#)
- [#unique\\_194/unique\\_194\\_Connect\\_42\\_subsection-im3 \[page 495\]](#)
- [#unique\\_194/unique\\_194\\_Connect\\_42\\_subsection-im4 \[page 495\]](#)
- [#unique\\_194/unique\\_194\\_Connect\\_42\\_subsection-im5 \[page 495\]](#)
- [#unique\\_194/unique\\_194\\_Connect\\_42\\_subsection-im6 \[page 496\]](#)
- [#unique\\_194/unique\\_194\\_Connect\\_42\\_subsection-im7 \[page 497\]](#)
- [#unique\\_194/unique\\_194\\_Connect\\_42\\_subsection-im8 \[page 497\]](#)
- [#unique\\_194/unique\\_194\\_Connect\\_42\\_subsection-im9 \[page 498\]](#)
- [#unique\\_194/unique\\_194\\_Connect\\_42\\_subsection-im10 \[page 499\]](#)

Hover over each shape for a description. Click the shape for more information.

## Define Metadata

Enter the general information or the metadata of the API:

- **swagger** - The version of the OpenAPI Specification. 2.0 being the latest version.
- **title** - A short summary of the API overall functionality.
- **description** -  
An expanded description of the API overall functionality and its usage specifics, if any.  
Descriptions can be entered in multiline. Basic html tags with appropriate attributes, can be used for styling.

Tags that can be used are:

- `<a>`
- `<b>`
- `<blockquote>`
- `<br>`
- `<cite>`
- `<code>`
- `<dd>`
- `<dl>`
- `<dt>`
- `<em>`
- `<i>`
- `<li>`
- `<ol>`
- `<p>`
- `<pre>`
- `<q>`
- `<small>`
- `<span>`
- `<strike>`
- `<strong>`
- `<sub>`
- `<sup>`
- `<u>`
- `<ul>`
- `<img>`

Any other tags will be auto-removed.

For the image, only Base64 encoded content is supported.

- **version** - The version of the API.

### Sample Code

API specification in YAML

```
swagger: '2.0'
info:
 title: SAP Workflow API
 description: |
```

```
This API provides functionality to work with
SAP Workflow Service.
You can, for example, start new workflow instances
and work with tasks.
version: 1.0
```

### Sample Code

API specification in JSON

```
{
 "swagger": "2.0",
 "info": {
 "title": "SAP Workflow API",
 "description": "This API provides functionality to work with SAP
Workflow Service.",
 "version": 1
 }
}
```

### Sample Code

API specification in YAML

```
swagger: '2.0'
info:
 title: SAP e-commerce API
 description: |
 This API provides functionality to build an
 e-commerce site selling electronic products
 The following scenarios are covered:
 * Customer can order products
 * Customer can provide product reviews
 * Retailer can create sales orders
 * Retailer can update product stock
version: 1.0
```

### Sample Code

API specification in JSON

```
{
 "swagger": "2.0",
 "info": {
 "title": "SAP e-commerce API",
 "description": "This API provides functionality to build an e-commerce
site selling electronic products.",
 "version": 1
 }
}
```

It is also a good practice to provide the license and the contacts details of the API. In the `license` object, you can provide a link to the licensing rules for the API, and in the `contact` object, you can provide details of the API provider.

### Sample Code

API specification in YAML

```
license:
 name: Apache-2.0
 url: "https://github.com/SAP/master/LICENSE"
contact:
 name: SAP API Business Hub team
 email: SAPAPIHubInfo@sap.com
 url: https://api.sap.com/#/community
```

### Sample Code

API specification in JSON

```
{
 "license": {
 "name": "Apache-2.0",
 "url": "https://github.com/SAP/master/LICENSE"
 },
 "contact": {
 "name": "SAP API Business Hub team",
 "email": "SAPAPIHubInfo@sap.com",
 "url": "https://api.sap.com/#/community"
 }
}
```

## Define Root URL

APIs have a root URL to which the paths or the endpoints are appended. The root URL is defined by `host`, `schemes`, and `basePath` on the root level of the OpenAPI Specification.:

- **host** - The host (name or IP address) serving the API. The host might include a port number. If a value for host is missing, then the host (including the port) providing the documentation is assumed also to provide the API.
- **schemes** - The transfer protocol expected by the API. Values must be either `http` or `https`. If a value for schemes is missing, then the scheme used to access the OpenAPI Specification definition becomes the scheme used to access the API.
- **basePath** - The base path on which the API is served, relative to the host. If this value is missing, then the API must be available directly under the host. The value must start with a leading forward slash (`/`).

### Sample Code

API specification in YAML

```
host: localhost
schemes:
 - https
basePath: /espm-cloud-web/espm.svc/secure
```

### Sample Code

API specification in JSON

```
{
 "host": "localhost",
```

```

"schemes": [
 "https"
],
"basePath": "/espm-cloud-web/espm.svc/secure"
}

```

## Add Attributes

You can define additional fields or attributes in the Open API Specification to enhance the usability of your API. Swagger JSON provides additional attributes that start with x-, such as x-servers. They can be used to describe additional functionality that is not covered by the standard OpenAPI Specification.

## Define Operations

Define the available paths (endpoints) and API operations. Paths are endpoints (resources) that your API exposes, such as `/products` or `/store/inventory`. Operations are the HTTP methods used to manipulate these paths, such as `put`, `get`, or `post`. Each operation is defined in its own `operation` object, including the path (endpoint), the HTTP method name, such as `get` or `put`, `summary`, and the `description`. You can add as many paths as you require.

### Sample Code

API specification in YAML

```

paths:
 /products:
 get:
 summary: Retrieves all products
 description: |
 Retrieves the list of all available
 products in the inventory.

```

### Sample Code

API specification in JSON

```

{
 "paths": {
 "/products": {
 "get": {
 "summary": "Retrieves all products",
 "description": "Retrieves the list of all available\nproducts in
the inventory.\n"
 }
 }
 }
}

```

All API paths or endpoints are relative to the root URL, for example, `/products` means `<scheme>://<host>/<basePath>/products`. That is, if you want to list all the products available in the inventory, your API call would be:

```
GET http://localhost//espm-cloud-web/espm.svc/secure/products
```

## Define Parameters

Define the input parameters that you want your API Operation to accept. The information about the input parameters is provided in the `Parameters` object of each operation in the API definition file. Each parameter

accepted by an operation is defined by a number of properties of the parameter object. The `in` property defines the location in which the parameter is passed.

### Sample Code

API specification in YAML

```
parameters:
 description: |
 The workflow instance ID for
 which task instances are returned.
 in: query
 name: workflowInstanceID
 required: true
 type: string
```

### Sample Code

API specification in JSON

```
{
 "parameters": {
 "description": "The workflow instance ID for which task instances are
returned.",
 "in": "query",
 "name": "workflowInstanceID",
 "required": true,
 "type": "string"
 }
}
```

For a detailed explanation about how to define the input parameters for your API, see [Adding Input Parameters - Headers and Queries \[page 500\]](#)

## Define Responses

Define the responses for API operations. For each possible response returned by an API operation, define a corresponding HTTP response status code and its description in the `responses` object. Defining HTTP status codes is crucial, as it describes the purpose of each method and the corresponding responses.

### Sample Code

API specification in YAML

```
responses:
 '204':
 description: |
 The task was successfully completed
 and the context was updated.
 '400':
 description: |
 Wrong format or structure
 of the provided request body.
 '403':
 description: |
 Access denied.
 You did not have the required permissions
 to access the resource.
 '404':
 description: |
 Not found. Possible reasons:
```



```

- You provided a wrong URL.
- The given task you are referring to doesn't
 exist.
- You are not allowed to access the task.
'422':
 description: |
 The workflow context provided in the
 request body contained invalid keys or values.
'500':
 description: |
 Internal server error.
 The operation you requested
 led to an error during execution.

```

## Sample Code

API specification in JSON

```

{
 "responses": {
 "204": {
 "description": "The task was successfully completed \nand the
context was updated.\n"
 },
 "400": {
 "description": "Wrong format or structure \nof the provided request
body.\n"
 }
 }
}

```

For a detailed explanation about how to define responses for your API Operation, see [Adding Responses \[page 508\]](#).

## Add Definitions

If the same data type, parameter or response are used in multiple operations within the same API or service, you can simplify the API definition file by creating a reusable definition for each of them, and reference it where relevant across the API.

## Add Security Definitions

Define all the implemented security mechanism for your APIs. Information about these schemes is provided under the `Security Definitions` object. The OpenAPI specification currently supports `basic`, `apiKey`, and `oauth2` security scheme types.

## Sample Code

```

securityDefinitions:
 basicAuthentication:
 type: basic
 oauth2:
 type: oauth2
 description: > To use this API, you need to obtain
the OAuth client credentials (client ID and secret)
using the SAP HANA BTP cockpit.
After that, pass these client credentials to
the SAP HANA BTP token endpoint
to obtain an access token.
 authorizationUrl: >-

```

```
https://host1/oauth/tok/v1?grant_type=client_credentials
flow: implicit
scopes:
 product_view: Read Product entities
 product_manage: Manage Product entities
```

## ↔ Sample Code

API specification in JSON

```
{
 "securityDefinitions": {
 "basicAuthentication": {
 "type": "basic"
 }
 }
}
```

For more information, see [Adding Security Definitions \[page 515\]](#).

## Add tags

To group related operations by categories, you can define tags to the operations so that they are rendered in an organized manner in the output. You define the tags under `tags` in the root swagger object. Each tag is identified by its `name`, which must be unique in the list of tags, and optional `description` and `externalDocs`. Then, you can assign the defined tags to operations, referring to them by names. Note that you can define a tag directly in an operation even if it is not defined on the root level.

## ↔ Sample Code

API specification in YAML

```
tags:
- name: Task Instances
 description: ''
- name: Workflow Definitions
 description: ''
- name: Workflow Instances
 description: ''
...
get:
 tags:
 - Task Instances
 description: Retrieves the context of a task.
```

## ↔ Sample Code

API specification in JSON

```
{
 "tags": [
 {
 "name": "Task Instances",
 "description": ""
 },
 {
 "name": "Workflow Definitions",
 "description": ""
 }
]
}
```

```

 "name": "Workflow Instances",
 "description": ""
 }
]
}
. .
{
 "get": {
 "tags": [
 "Task Instances"
],
 "description": "Retrieves the context of a task."
 }
}

```

## Add External Links

Add external links, if any, in your API Definition to provide enhanced user assistance for using the APIs. You define the external links in the `externalDocs` object, which can be used on the root level for the entire API, and/or on the operation or tag level, as required.

### Sample Code

API specification in YAML

```

externalDocs:
 description: SAP Workflow Documentation
 url: https://help.sap.com/viewer/p/WORKFLOW_SERVICE

```

### Sample Code

API specification in JSON

```

{
 "externalDocs": {
 "description": "SAP Workflow Documentation",
 "url": "https://help.sap.com/viewer/p/WORKFLOW_SERVICE"
 }
}

```

Once you have defined all the necessary information of the API, choose **File > Save** in API designer .

### Note

If you have created a new API, enter a name for it. If you are editing an existing API, save it under the existing name. When you save, you may receive warning messages about missing parameters. In this case, add the required parameters and save again.

## Related Information

[Perform Additional Tasks in API Designer \[page 532\]](#)

## 1.5.2.2.1 Adding Input Parameters - Headers and Queries

Define all input parameters for your API operation, irrespective of whether they are mandatory or optional. The parameters information can be modeled under `parameters` definitions object in the OpenAPI specification

In a RESTful API, the input parameters can be any of the following types:

- **Query Parameters**

When submitting a request, these parameters form the query string part at the end of the request URL. A question mark (?) character delimits the URL from the query string. For example, `Products?top=2`. Multiple parameters can be supplied as name/value pairs in the format `name="value"`. Each name/value pair is separated by the ampersand (&) character. For example, `Products?skip=10&top=2&someName="someValue"`.

### Sample Code

API specification in YAML

```
parameters:
 - in: query
 name: pageNumber
 type: integer
 description: The page number from which the items must be
listed .
 - in: query
 name: limit
 type: pageSize
 description: The number of items to be returned on the specified
page.
```

In the above example a GET call, say `GET /products?pageNumber=2&pageSize=20` returns 20 products from page number 2.

### Sample Code

API specification in JSON

```
{
 "parameters": [
 {
 "in": "query",
 "name": "pageNumber",
 "type": "integer",
 "description": "The page number from which the items must be
listed ."
 },
 {
 "in": "query",
 "name": "limit",
 "type": "pageSize",
 "description": "The number of items to be returned on the
specified page."
 }
]
}
```

In the above example a GET call, say `GET /products?pageNumber=2&pageSize=20` returns 20 products from page number 2.

- **Header Parameters**

Header parameters are components of the header section of an HTTP request or response. The name of a header field must be immediately followed by a colon : character. Any whitespace between the field name and the colon is syntactically incorrect. The header parameter value is provided as a plain text string. For example, `Accept: application/json`.

For example, suppose a call to `GET /check` requires the `Request-ID` header:

#### Sample Code

```
GET /check HTTP/1.1
Host: localhost
Request-ID: 2345-23567-4356-ab32-43ed
```

In the API designer, you will write the operation definition as follows:

#### Sample Code

API specification in YAML

```
/check:
 get:
 summary: Checks if the server is up.
 parameters:
 - in: header
 name: Request-ID
 type: string
 required: true
 responses:
 200:
 description: OK
```

#### Sample Code

API specification in JSON

```
{
 "/check": {
 "get": {
 "summary": "Checks if the server is up.",
 "parameters": [
 {
 "in": "header",
 "name": "Request-ID",
 "type": "string",
 "required": true
 }
],
 "responses": {
 "200": {
 "description": "OK"
 }
 }
 }
 }
}
```

- **Path Parameters**

Path parameters are a flexible way of parameterizing the actual values used when creating the path to a resource. For example, if the value of the Product ID needs to be present in the path name, then this can be parameterized as follows: `/Products/{ProductId}`.

## Note

The parameter name must be the same as specified in the path. Also, remember to add `required: true`, because path parameters are always required.

## Sample Code

API specification in YAML

```
paths:
 /products{productId}:
 get:
 parameters:
 - in: path
 name: productId # Note the name is the same as in the path
 required: true
 type: integer
 minimum: 1
 description: The product ID.
 responses:
 200:
 description: OK
```

## Sample Code

API specification in JSON

```
{
 "paths": {
 "/products{productId}": {
 "get": {
 "parameters": [
 {
 "in": "path",
 "name": "productId",
 "required": true,
 "type": "integer",
 "minimum": 1,
 "description": "The product ID."
 }
],
 "responses": {
 "200": {
 "description": "OK"
 }
 }
 }
 }
 }
}
```

You can familiarize more about defining parameters with the below examples:

The following APIs use OData protocol, and they support OData system queries. The commonly used system query parameters are defined in the following example:

## Sample Code

API Specification in YAML

```
parameters:
```

```

top:
 name: $top
 in: query
 description: >-
 Show only the first N elements, where N is a positive integer.
 If a value less than 0 is specified, the URI should be considered
malformed.
 ref [link](http://www.odata.org/documentation/odata-version-2-0/uri-
conventions/#SystemQueryOptions) for more information
 type: integer
 minimum: 0
skip:
 name: $skip
 in: query
 description: >-
 Skip the first N elements, where N is a positive integer as specified
by this query option.
 If a value less than 0 is specified, the URI should be considered
malformed.
 ref [link](http://www.odata.org/documentation/odata-version-2-0/uri-
conventions/#SystemQueryOptions) for more information
 type: integer
 minimum: 0
count:
 name: $count
 in: query
 description: >-
 Include count of elements.
 ref [link](http://www.odata.org/documentation/odata-version-2-0/uri-
conventions/#SystemQueryOptions) for more information
 type: boolean

```

## ↔ Sample Code

### API Specification in JSON

```

{
 "parameters": {
 "top": {
 "name": "$top",
 "in": "query",
 "description": "Show only the first N elements, where N is a
positive integer.",
 "type": "integer",
 "minimum": 0
 },
 "skip": {
 "name": "$skip",
 "in": "query",
 "description": "Skip the first N elements, where N is a positive
integer as specified by this query option.",
 "type": "integer",
 "minimum": 0
 },
 "count": {
 "name": "$count",
 "in": "query",
 "description": "Include count of elements.",
 "type": "boolean"
 }
 }
}

```

These parameters can be referred in the `get` operation listed under [▶ paths ▶ /Products ▶](#) as follows:

## Sample Code

### API Specification in YAML

```
paths:
 /Products:
 get:
 summary: Get entities from entity set Products
 description: Get entities from entity set Products
 security:
 - oauth2:
 - product_view
 tags:
 - Products
 parameters:
 - $ref: '#/parameters/top'
 - $ref: '#/parameters/skip'
 - $ref: '#/parameters/count'
```

## Sample Code

### API Specification in JSON

```
{
 "paths": {
 "/Products": {
 "get": {
 "summary": "Get entities from entity set Products",
 "description": "Get entities from entity set Products",
 "security": [
 {
 "oauth2": [
 "product_view"
]
 }
],
 "tags": [
 "Products"
],
 "parameters": [
 {
 "$ref": "#/parameters/top"
 },
 {
 "$ref": "#/parameters/skip"
 },
 {
 "$ref": "#/parameters/count"
 }
]
 }
 }
 }
}
```

The advantage of defining the parameters in this way is that they can be defined once, and then later referenced from multiple places within the same specification file.

In the following example, other system queries such as `$select`, `$expand` and `$orderby` are defined at the operations level:



## Sample Code

### API Specification in YAML

```
paths:
 /Products:
 get:
 summary: Get entities from entity set Products
 description: Get entities from entity set Products
 security:
 - oauth2:
 - product_view
 tags:
 - Products
 parameters:
 - $ref: '#/parameters/top'
 - $ref: '#/parameters/skip'
 - $ref: '#/parameters/count'
 - name: $orderby
 in: query
 description: >-
 Order by property values, for example `?$orderby=Name` for sorting
 the Products by Name and `?$orderby=Name desc` for sorting the
 Products by Name in descending order
 type: array
 uniqueItems: true
 items:
 type: string
 enum:
 - Category
 - Category desc
 - CategoryName
 - name: $select
 in: query
 description: >-
 Select properties to be returned, The value of a $select System
 Query Option is a comma-separated list of selection clauses. Each
 selection clause may be a Property name, Navigation Property name.
 for example `?$select=Category,Name` so that only Category and
 Name is returned
 type: array
 uniqueItems: true
 items:
 type: string
 enum:
 - Category
 - CategoryName
 - CurrencyCode
 - name: $expand
 in: query
 description: >-
 Expand related entities, The syntax of a $expand query option is a
 comma-separated list of Navigation Properties. for example
 `?$expand=Supplier` to get the related Supplier information
 inline.
 type: array
 uniqueItems: true
 items:
 type: string
 enum:
 - CustomerReview
 - Supplier
```

## Sample Code

### API Specification in JSON

```
{
 "paths": {
 "/Products": {
 "get": {
 "summary": "Get entities from entity set Products",
 "description": "Get entities from entity set Products",
 "security": [
 {
 "oauth2": [
 "product_view"
]
 }
],
 "tags": [
 "Products"
],
 "parameters": [
 {
 "$ref": "#/parameters/top"
 },
 {
 "$ref": "#/parameters/skip"
 },
 {
 "$ref": "#/parameters/count"
 },
 {
 "name": "$orderby",
 "in": "query",
 "description": "Order by property values",
 "type": "array",
 "uniqueItems": true,
 "items": {
 "type": "string"
 },
 "enum": [
 "Category",
 "Category desc",
 "CategoryName"
]
 },
 {
 "name": "$select",
 "in": "query",
 "description": "Select properties to be returned",
 "type": "array",
 "uniqueItems": true,
 "items": {
 "type": "string"
 },
 "enum": [
 "Category",
 "CategoryName",
 "CurrencyCode"
]
 },
 {
 "name": "$expand",
 "in": "query",
 "description": "Expand related entities",
 "type": "array",
 "uniqueItems": true,
 "items": {
```

```

 "type": "string"
 },
 "enum": [
 "CustomerReview",
 "Supplier"
]
}
]
}
}
}
}
}
}
}
}
}
}

```

In the following example, a paths parameter called `ProductId` is defined as a required field:

## Sample Code

API Specification in YAML

```

paths:
 /Products{ProductId}:
 get:
 summary: Get entity from Products by key.
 description: Returns the entity with the key from Products
 tags:
 - Products
 parameters:
 - name: ProductId
 in: path
 required: true
 description: 'key: ProductId'
 type: string

```

## Sample Code

API Specification in JSON

```

{
 "paths": {
 "/Products{ProductId}": {
 "get": {
 "summary": "Get entity from Products by key.",
 "description": "Returns the entity with the key from Products",
 "tags": [
 "Products"
],
 "parameters": [
 {
 "name": "ProductId",
 "in": "path",
 "required": true,
 "description": "key: ProductId",
 "type": "string"
 }
]
 }
 }
 }
}

```

## Related Information

[Parameters Definitions Object](#) ↗

### 1.5.2.2.2 Adding Responses

Define all the expected responses, including the response headers, response message, and error messages.

The server receives an incoming request and responds with a message indicating whether the request was successful or not. Such a response consists of the following elements:

- An HTTP status code
- Zero or more Response Headers
- An optional Response Body

Use the standard HTTP status codes for describing the success or failure of the request. It is also recommended to use the HTTP response headers.

## Related Information

[Adding Responses without a Body \[page 508\]](#)

[Responses with a Body \[page 509\]](#)

[Response Headers \[page 511\]](#)

[Error Response \[page 512\]](#)

#### 1.5.2.2.2.1 Adding Responses without a Body

It is a valid and common feature of a RESTful API that the response to certain operations contains no body. For instance, the Update operations (implemented by the HTTP method PUT ) belonging to the ESPM OData API in the example below typically return an HTTP status code 204 with no message body.

In the example below, the same information is modeled using the Open API specification:

#### Sample Code

API specification in YAML

```
paths:
 '/Products('{ProductId}')':
 put:
 summary: Update a product entity in Products entityset
 description: Update a product entity in Products entityset
 tags:
 - Products
 parameters:
 - name: ProductId
```

```

 in: path
 required: true
 description: 'key: ProductId'
 type: string
 - name: Product
 in: body
 description: The entity to patch
 schema:
 $ref: '#/definitions/Product'
 responses:
 '204':
 description: Empty response

```

## Sample Code

API specification in JSON

```

{
 "paths": {
 "/Products('{ProductId}')": {
 "put": {
 "summary": "Update a product entity in Products entityset",
 "description": "Update a product entity in Products entityset",
 "tags": [
 "Products"
],
 "parameters": [
 {
 "name": "ProductId",
 "in": "path",
 "required": true,
 "description": "key: ProductId",
 "type": "string"
 },
 {
 "name": "Product",
 "in": "body",
 "description": "The entity to patch",
 "schema": {
 "$ref": "#/definitions/Product"
 }
 }
],
 "responses": {
 "204": {
 "description": "Empty response"
 }
 }
 }
 }
 }
}

```

### 1.5.2.2.2.2 Responses with a Body

For certain operations like Create (implemented by the HTTP method POST), the newly created resource is typically returned as the response body. This is to save the client from then having to perform a subsequent Read operation to determine the exact server-side state of the newly created resource.

In the example below, this information is modeled using the Open API specification. Since the definition of the Product returned in the response is exactly the same as that previously defined in the section on requests, the reference to the Product definition can be reused

## Sample Code

API specification in YAML

```
paths:
 '/Products('{ProductId}')':
 post:
 summary: Create a product
 description: Create a product entity in Products entityset
 tags:
 - Products
 parameters:
 - name: Product
 in: body
 description: Product entity to be created
 schema:
 $ref: '#/definitions/Product'
 responses:
 '201':
 description: Created product
 schema:
 $ref: '#/definitions/Product'
```

## Sample Code

API specification in JSON

```
{
 "paths": {
 "/Products('{ProductId}')": {
 "post": {
 "summary": "Create a product",
 "description": "Create a product entity in Products entityset",
 "tags": [
 "Products"
],
 "parameters": [
 {
 "name": "Product",
 "in": "body",
 "description": "Product entity to be created",
 "schema": {
 "$ref": "#/definitions/Product"
 }
 }
],
 "responses": {
 "201": {
 "description": "Created product",
 "schema": {
 "$ref": "#/definitions/Product"
 }
 }
 }
 }
 }
 }
}
```

## 1.5.2.2.2.3 Response Headers

Along with the response message, the server can respond with zero or more headers.

In the example below, the custom HTTP header DataServiceVersion is described. This header informs the client about the version of OData used to build the response.

### Sample Code

API specification in YAML

```
paths:
 '/Products('{ProductId}')':
 post:
 summary: Create a new product
 description: Post a new entity to entity set Products
 tags:
 - Products
 parameters:
 - name: Product
 in: body
 description: Product entity to be created
 schema:
 $ref: '#/definitions/Product'
 responses:
 '201':
 description: Created Product
 headers:
 DataServiceVersion:
 type: string
 description: |
 The value states the OData version the server used to
 generate the response
 and that should be used by the client to determine if it can
 correctly interpret the response
 schema:
 $ref: '#/definitions/Product'
```

### Sample Code

API specification in JSON

```
{
 "paths": {
 "/Products('{ProductId}')": {
 "post": {
 "summary": "Create a new product",
 "description": "Post a new entity to entity set Products",
 "tags": [
 "Products"
],
 "parameters": [
 {
 "name": "Product",
 "in": "body",
 "description": "Product entity to be created",
 "schema": {
 "$ref": "#/definitions/Product"
 }
 }
],
 "responses": {
 "201": {
```

```

 "description": "Created Product",
 "headers": {
 "DataServiceVersion": {
 "type": "string",
 "description": "The value states the OData version
the server used to generate the response \nand that should be used by the
client to determine if it can correctly interpret the response\n"
 }
 },
 "schema": {
 "$ref": "#/definitions/Product"
 }
 }
 }
 }
}

```

Based on such a definition, the client can then expect to receive a response containing at least the `DataServiceVersion` header. For example, a Gateway OData server returns the following headers. Notice that `DataServiceVersion` is among those headers.

#### ↔ Sample Code

```

cache-control:no-store, no-cache, must-revalidate, max-age=0, post-check=0,
pre-check=0
content-encoding:gzip
content-length:785
content-type:application/xml
dataserviceversion:2.0
pragma:no-cache
sap-metadata-last-modified:Fri, 15 Jan 2016 04:01:16 GMT
server:SAP NetWeaver Application Server / ABAP 702

```

#### 📌 Note

Notice also that all the HTTP header fields returned from an ABAP OData server are in lowercase. This conforms to the HTTP standard that all HTTP header fields are case insensitive.

## 1.5.2.2.2.4 Error Response

In case of an error, the server should return an error message that best describes the cause of the problem and, where possible, provides information on how to correct that problem. It is a good practice to use standard HTTP error status codes to inform about the nature of the error and have a common error schema describing each of these status codes.

The text of error message should always help move the user towards the solution. Simply responding with a generic error message such as "Internal server error" leaves the user stranded and unable to proceed towards a solution.

For example, the ESPM OData API returns errors using the OData error format, and therefore we have documented some of the commonly returned error codes. Since the error messages are the same for all the operations, we have documented them in one place using the responses object of the Open API specification, then referenced those definitions within the same file.



## Sample Code

API specification in YAML

```
responses:
 401:
 description: Unauthorized
 404:
 description: Not Found
 schema:
 $ref: '#/definitions/odata.error'
 400:
 description: Bad Request
 schema:
 $ref: '#/definitions/odata.error'
 500:
 description: Internal server error
 schema:
 $ref: '#/definitions/odata.error'
definitions:
 odata.error:
 type: object
 properties:
 code:
 type: string
 description: Error code
 message:
 type: object
 properties:
 lang:
 type: string
 description: Language code of the error message
 value:
 type: string
 description: Detailed error message
```

## Sample Code

API specification in JSON

```
{
 "responses": {
 "400": {
 "description": "Bad Request",
 "schema": {
 "$ref": "#/definitions/odata.error"
 }
 },
 "401": {
 "description": "Unauthorized"
 },
 "404": {
 "description": "Not Found",
 "schema": {
 "$ref": "#/definitions/odata.error"
 }
 },
 "500": {
 "description": "Internal server error",
 "schema": {
 "$ref": "#/definitions/odata.error"
 }
 }
 },
 "definitions": {
```



## Sample Code

API specification in JSON

```
{
 "paths": {
 "/Products('{ProductId}')": {
 "put": {
 "summary": "Update entity in EntitySet Products",
 "description": "Update entity in EntitySet Products",
 "tags": [
 "Products"
],
 "parameters": [
 {
 "name": "ProductId",
 "in": "path",
 "required": true,
 "description": "key: ProductId",
 "type": "string"
 },
 {
 "name": "Product",
 "in": "body",
 "description": "The entity to patch",
 "schema": {
 "$ref": "#/definitions/Product"
 }
 }
]
 },
 "responses": {
 "204": {
 "description": "Empty response"
 },
 "400": {
 "$ref": "#/responses/400"
 },
 "401": {
 "$ref": "#/responses/401"
 },
 "404": {
 "$ref": "#/responses/404"
 },
 "500": {
 "$ref": "#/responses/500"
 }
 }
 }
 }
}
```

### 1.5.2.2.2.3 Adding Security Definitions

Security definitions allow you to define various authentication types (security schemes) supported by an API. If you want to implement any authentication mechanism for your API, then we recommended you to define and apply the associated authentication types in your API.

OpenAPI specification lets you define the following authentication types for an API:

- **basic**  
Used in situations where simple userid/password based authentication is sufficient
- **apiKey**  
Used in situations where the application must authenticate itself by presenting an API Key value as part of every request. This type of authentication is independent of any user authentication.
- **oauth2**  
Used when the OAuth2 based authentication is required.

Authentication types are described using the `securityDefinitions` and `security` properties of the OpenAPI specification. Use the `securityDefinitions` property to define all authentication types supported by the API, and then use the `security` property to apply specific authentication types to the whole API.

In the API below, it supports three security schemes named `basicAuthentication`, `apiKeyAuth`, and `OAuth2`. These names are used to refer to these security schemes from elsewhere within the API.

### Sample Code

API specification in YAML

```
info:
 title: >-
 This sample is a reference service, showcases the e-commerce APIs for
 products, and suppliers.
 securityDefinitions:
 basicAuthentication:
 type: basic
 apiKeyAuth:
 type: apiKey
 in: header
 name: X-API-Key
 OAuth2:
 type: oauth2
 description: >
 To use this REST API, you need to get OAuth client credentials (client
 ID
 and secret) using the SAP BTP Cockpit. After that,
 you need to pass the obtained client credentials to the SAP BTP token
 endpoint to obtain an access token.
 tokenUrl: >-
 https://api.hana.ondemand.com/oauth2/apitoken/v1?
 grant_type=client_credentials
 flow: application
 scopes:
 product_view: Read Product entities
 product_manage: Manage Product entities
```

### Sample Code

API specification in JSON

```
{
 "info": {
 "title": "This sample is a reference service, showcases the e-
 commerce APIs for products, and suppliers."
 },
 "securityDefinitions": {
 "basicAuthentication": {
 "type": "basic"
 },
 "apiKeyAuth": {
 "type": "apiKey",
 "in": "header",
```

```

 "name": "X-API-Key"
 },
 "OAuth2": {
 "type": "oauth2",
 "description": "To use this REST API, you need to get OAuth
client credentials (client ID and secret) using the SAP BTP cockpit. After
that, you need to pass the obtained client credentials to the SAP BTP token
endpoint to obtain an access token. \n",
 "tokenUrl": "https://api.hana.ondemand.com/oauth2/apitoken/v1?
grant_type=client_credentials",
 "flow": "application",
 "scopes": {
 "product_view": "Read Product entities",
 "product_manage": "Manage Product entities"
 }
 }
 }
}

```

After you have defined the security schemes in the `securityDefinitions` property, you can apply them to the whole API using the `security` property on the root level. When used on the root level, `security` applies the defined security definitions globally to all the operations within the API. In the following example, the API calls can be authenticated using either basic authentication (username and password) or API key.

### Sample Code

API specification in YAML

```

info:
 title: >-
 This sample is a reference service that showcases the e-commerce APIs for
products, and suppliers.
securityDefinitions:
 basicAuthentication:
 type: basic
 apikeyAuth:
 type: apiKey
 in: header
 name: X-API-Key
 OAuth2:
 type: oauth2
 description: >
 To use this REST API, you need to get OAuth client credentials (client
ID
and secret) using the SAP BTP cockpit. After that,
you need to pass the obtained client credentials to the SAP BTP token
endpoint to obtain an access token.
 tokenUrl: >-
 https://api.hana.ondemand.com/oauth2/apitoken/v1?
grant_type=client_credentials
 flow: application
 scopes:
 product_view: Read Product entities
 product_manage: Manage Product entities
 security:
 - basicAuthentication: []
 - apikeyAuth: []

```

### Sample Code

API specification in JSON

```
{
```

```

 "info": {
 "title": "This sample is a reference service, which showcases the e-commerce APIs for products, and suppliers."
 },
 "securityDefinitions": {
 "basicAuthentication": {
 "type": "basic"
 },
 "apiKeyAuth": {
 "type": "apiKey",
 "in": "header",
 "name": "X-API-Key"
 },
 "OAuth2": {
 "type": "oauth2",
 "description": "To use this REST API, you need to get OAuth client credentials (client ID and secret) using the SAP BTP cockpit. After that, you need to pass the obtained client credentials to the SAP BTP token endpoint to obtain an access token. \n",
 "tokenUrl": "https://api.hana.ondemand.com/oauth2/apitoken/v1?grant_type=client_credentials",
 "flow": "application",
 "scopes": {
 "product_view": "Read Product entities",
 "product_manage": "Manage Product entities"
 }
 }
 },
 "security": [
 {
 "basicAuthentication": []
 },
 {
 "apiKeyAuth": []
 }
]
 }
}

```

The security scheme applied at the root level can be overridden in individual operations. In the following example, `basic` is defined and applied as security scheme at the root level. Whereas, for `/products`, the security scheme defined at the root level is overridden by having no security scheme .

## 🔗 Sample Code

API specification in YAML

```

securityDefinitions:
 basicAuthentication:
 type: basic
To apply Basic auth to the whole API:
security:
 - basicAuthentication: []
paths:
 /something:
 get:
 summary: Get entities from entity set Products
 description: Get entities from entity set Products
 # To apply Basic auth to an individual operation:
 security: [] #No security

 responses:
 200:
 description: OK (successfully authenticated)

```

## Sample Code

API specification in JSON

```
{
 "securityDefinitions": {
 "basicAuthentication": {
 "type": "basic"
 }
 },
 "security": [
 {
 "basicAuthentication": []
 }
],
 "paths": {
 "/something": {
 "get": {
 "summary": "Get entities from entity set Products",
 "description": "Get entities from entity set Products",
 "security": [],
 "responses": {
 "200": {
 "description": "OK (successfully authenticated)"
 }
 }
 }
 }
 }
}
```

### 1.5.2.2.2.4 Additional Attributes in OpenAPI Specification

OpenAPI Specification allows you to define additional attributes or custom extensions that start with `x-`, such as `x-servers`.

You can use an OpenAPI Specification to describe extra functionality of the API that is not covered by the standard OpenAPI Specification.

Additional attributes or custom extensions are supported on the `root` level of the OpenAPI Specification and in the following places:

- `info` section
- `paths` section
- `responses` section
- `tags`
- Security schemes

The following additional attributes have been introduced for use with API designer. All these attributes must be defined at the root level of the OpenAPI Specification.

#### List of additional attributes

## x-sap-api-type

### Sample Code

```
{
 "x-sap-api-type": "ODATA"
}
```

You can use this attribute to display the API type. For example:

The API Type is set according to the following rules of precedence:

1. If EDMX file is present, then the type is set as ODATA.
2. If WSDL file is present, then the type is set as SOAP.
3. If x-sap-api-type is mentioned in the swagger, then the respective value is set. For OData APIs, to differentiate the V2, and V4 OData APIs, you will need to specify the following values in swagger for the x-sap-api-type attribute:

Type of API	Input
OData V2 API	ODATA
OData V4 API	ODATAV4

4. The default is REST.

## x-sap-shortText

The x-sap-shortText attribute is used to display a short description of your APIs.

### Note

The x-sap-shortText is a mandatory attribute that must be defined in the API definition file.

### Sample Code

```
{
 "info": {
 "title": "SAP Workflow service API",
 "description": "Provides functionality to work with SAP Workflow service.
You can, for example, start new workflow instances and work with tasks."
 },
 "x-sap-shortText": " Work with the SAP Workflow service."
}
```

The text should not exceed 180 characters. The following characters are allowed in the x-sap-shortText:

Character	Description
A-Z	Latin letters, case insensitive



Character	Description
0-9	Numbers
	Space
-	Underscore
- - -	Different Hyphens
.	Dot
,	Comma
()	Paranthesis
's	Possession apostrophe

Make sure a short text doesn't contain special characters like \$, &, !, %, \, /, \*, ; and so forth.

Do not use semicolons to separate two sentences. Rather, make it two separate sentences.

Please note, though forward slash is not an allowed character to use, it is allowed in a product name, such as S/4HANA or combination of words country/region.

## x-sap-stateInfo

You use `x-sap-stateInfo` attribute to display the current status of an API.

### Note

The `x-sap-stateInfo` is an optional attribute. That is, if you do not use this attribute in your API definition file, then by default, the current status of an API is marked as `Active`. However, If you want to publish your API in the `Beta` status or you have decided to transition your API from `Active` to `Deprecated` or `Decommissioned` status, then it is mandatory to use this attribute to indicate the new status of your API.

## API status

An API can be marked with any of the following statuses:

- Beta

The following is a sample code snippet of an API definition file in which `x-sap-stateInfo` attribute is used to indicate that the current status of the API is `Beta`.

### Sample Code

```
{
 "swagger": "2.0",
 "info": {
 "title": "Business API",
 "version": "1.1.0",
 "description": "API for Reading Business Partner, Supplier,
Customer and Contact Persons"
 },
 "x-sap-stateInfo": {
 "state": "Beta"
 }
}
```

```
}
}
```

- Active  
If `x-sap-stateInfo` attribute is not defined or left empty, then the default status of the API is taken as Active.
- Deprecated  
The following is a sample code snippet of an API definition file in which `x-sap-stateInfo` attribute is used to indicate that the current status of the API is deprecated as on 14th August 2018, and a new version of the API is available:

#### Sample Code

```
{
 "swagger": "2.0",
 "info": {
 "title": "Business API",
 "version": "1.1.2",
 "description": "API for Reading Business Partner, Supplier,
Customer and Contact Persons"
 },

 "x-sap-stateInfo": {
 "state": "Deprecated",
 "deprecationDate": "2018-08-14",
 "successorApi": "http://api.sap.com/api/
product_text_classification_api"
 }
}
```

#### Note

You must enter the `deprecationDate` in the format `yyyy-mm-dd`

- Decommissioned  
The following is a sample code snippet of an API definition file in which `x-sap-stateInfo` attribute is used to indicate that the current status of the API is decommissioned as on 05th September 2018, and a new version of the API is available:

#### Sample Code

```
{
 "swagger": "2.0",
 "info": {
 "title": "Business API",
 "version": "1.1.2",
 "description": "API for Reading Business Partner, Supplier,
Customer and Contact Persons"
 },

 "x-sap-stateInfo": {
 "state": "Decommissioned",
 "decommissionedDate": "2018-10-05",
 "successorApi": "http://api.sap.com/api/
product_text_classification_api"
 }
}
```

## Change Log

If you have defined the `x-sap-stateInfo` attribute, then you must also ensure that you have entered the change log information in the `Artifact.json` file of your API package.

### Note

If you do not enter the changelog information in `Artifact.json` file, then it results in unsuccessful builds.

**Change Log for a deprecate API:** The following is a sample code snippet of the `Artifact.json` file in which `changelog` attribute is used to indicate the most recent state of the API.

### Sample Code

```
{
 "type": "API",
 "changelog": [
 {
 "state": "Deprecated",
 "date": "2018-09-14",
 "version": "1.0",
 "notes": "New api with enhanced functionality is available"
 },
 {
 "state": "Active",
 "date": "2018-01-18",
 "version": "1.0.0",
 "notes": "Some bug fixes and performance enhancement"
 },
 {
 "state": "Active",
 "date": "2021-01-08",
 "version": "1.0.0",
 "notes": "Notifications API has been moved out of Equipment
API. Visit
Notifications API"
 }
]
}
```

### Note

You must enter the `date` in the format `yyyy-mm-dd`. The most recent state of the API must appear as the first entry under `changelog`, and the values defined for the `state` attribute in `API definition` file and `Artifact.json` file must be same. It is a good practice to indicate the recent changes made to the API using the `notes` attribute. This will help your API consumers to know if they need to follow certain rules or conditions before using the API.

The following images show a sample API, which is marked `Deprecated` on the :

The screenshot displays the SAP API Management console for a deprecated API. At the top, there is a warning icon and the word "Deprecated" in orange. Below this, the title "Deprecated API" is centered, followed by the description "API for Reading Business Partner, Supplier, Customer and Contact Persons". Two buttons, "Show API Key" and "Download SDK", are visible. A navigation bar at the bottom shows "API References" and "Details" (which is selected). Below the navigation bar, the "API Info" section provides details: Version: 1.0, Status: ⚠ Deprecated, Last Modified: 14 Aug 2018, and Deprecated: 14 Aug 2018. It also lists the Successor API as "Product Text Classification API" and includes a "Download Specification" button. The Production URL is `http://host:8080/sap/opu/odata/sap/API_BUSINESS_PARTNER/$metadata` with a "More" link. Documentation is linked to "Business documentation".

### Change Log

Version	Status	Modified	Notes
1.1.2	Deprecated	14 Aug 2018	This api has been deprecated

**Change Log for a decommissioned API** The following is a sample code snippet of the `Artifact.json` file in which `changeLog` attribute is used to indicate the decommissioned state of the API.

#### 📌 Note

When you transition an API from one state to another, you must enter the changelog information in `Artifact.json` file. SAP Content pipeline strongly recommends the following:

- You add the changelog information if you are publishing the first or initial version of an API.
- You add the changelog information for every consecutive update of an already published API.

#### 📄 Sample Code

```
{
 "type": "API",
 "changeLog": [
 {
```

```
 "state": "Decommissioned",
 "date": "2018-10-05",
 "version": "1.1.2",
 "notes": "This API is decommissioned"
 },
 {
 "state": "Deprecated",
 "date": "2018-07-18",
 "version": "1.0",
 "notes": "This API is deprecated"
 },
 {
 "state": "Active",
 "date": "2018-01-18",
 "version": "1.0.0",
 "notes": "Some bug fixes and performance enhancement"
 }
]
}
```

### 📌 Note

You must enter the date in the format `yyyy-mm-dd`. The most recent recent state of the API must appear as the first entry under `changelog`, and the values defined for the `state` attribute in `API definition` file and `Artifact.json` file must be same. It is a good practice to indicate the recent changes made to the API using the `notes` attribute. This will help your API consumers to know if they need to follow certain rules or conditions before using the API.

The following images shows a sample API, which is marked `Decommissioned` on the :

The screenshot shows a web interface for a decommissioned API. At the top, there is a red warning icon and the text "Decommissioned". Below this, the title "Decommissioned API" is displayed, followed by the description "API for Reading Business Partner, Supplier, Customer and Contact Persons". There are two buttons: "Show API Key" and "Download SDK". A navigation bar at the bottom of the main content area shows "API References" and "Details" (which is selected). Below the navigation bar, the "Details" section shows the following information:

- Version: 1.1.2
- Status: ⚠ Decommissioned
- Last Modified: 05 Sep 2018
- Decommissioned: 05 Sep 2018
- Successor API: [Product Text Classification API](#)
- API for Reading Business Partner, Supplier, Customer and Contact Persons
- [Download Specification](#)
- Production URL: [http://host:8080/sap/opu/odata/sap/API\\_BUSINESS\\_PARTNER/\\$metadata](http://host:8080/sap/opu/odata/sap/API_BUSINESS_PARTNER/$metadata)
- Documentation: [Business documentation](#)

Below the details section is a "Change Log" section with a table:

Version	Status	Modified	Notes
1.1.2	Decommissioned	05 Sep 2018	This api is decommissioned
1.1.2	Deprecated	18 Jul 2017	This api is deprecated
1.1.2	Active	18 Jul 2016	Now user can do put as well
1.1.1	Active	18 Jul 2017	Now user can do delete call

## Deprecating API Operation or Parameter

### Marking an API operation or parameter as deprecated

#### ⓘ Note

In OpenAPI 2.0 specification, you have the provision to mark only an API operation as deprecated whereas in OpenAPI 3.0 specification, you can mark both API operation and a parameter as deprecated.

In OpenAPI 2.0, use `deprecated: true` to mark an API operation as deprecated.

#### 🔗 Sample Code

operation deprecation

```
{
 "paths": {
 "/list": {
 "get": {
 "responses": {
```

```

 "200": {
 "description": "This API Operation is deprecated."
 }
 },
 "deprecated": true
}
}
}

```

In OpenAPI 3.0, use `deprecated: true` to mark an API operation and parameter as deprecated.

#### Sample Code

operation deprecation

```

{
 "paths": {
 "/list": {
 "get": {
 "responses": {
 "200": {
 "description": "This API Operation is deprecated."
 }
 }
 },
 "deprecated": true
 }
 }
}

```

#### Sample Code

parameter deprecation

```

{
 "in": "query",
 "name": "id",
 "required": true,
 "schema": {
 "type": "string"
 },
 "deprecated": true,
 "description": "Deprecated, use 'itemId' parameter instead."
}

```

For more information, see [API Deprecation Policy](#).

## x-sap-csrf-token-path

You use this attribute to provide a path relative to the basepath of your API for fetching X-CSRF-Token. That is, this attribute must contain the path of a resource that handles the fetching of x-csrf token requests. The relative path provided must not include server and transfer protocol information.

#### Sample Code

```

{

```

```

"host": "api.workflow.com",
"schemes": [
 "https"
],
"basePath": "/workflow-service/rest",
"x-sap-csrf-token-path": "/<relative_path_for_handling_csrf_token_fetch>"
}

```

### Note

The relative\_path must be a path relative to the basePath that you have provided in your API.

### Sample Code

Example

```

{
 "host": "api.workflow.com",
 "schemes": [
 "https"
],
 "basePath": "/workflow-service/rest",
 "x-sap-csrf-token-path": "/v1/xsrf-token"
}

```

## x-sap-software-min-version

You use this attribute to provide the minimum software version. For example:

### Sample Code

```

{
 "x-sap-software-min-version": "SAP NetWeaver 2.0"
}

```

## x-sap-ext-overview

You can use this attribute to provide stakeholder-specific information. For example:

### Sample Code

```

{
 "x-sap-ext-overview": [{
 "name": "Communication Scenarios",
 "values": ["SAP_COM_0025 Name of SAP_COM_0025", "SAP_COM_0028 Name of SAP_COM_00028"]
 },
 {
 "name": "Additional Property",
 "values": ["EntryValue1", "EntryValue2", "EntryValue3"]
 }
]
}

```



```
}
```

### → Remember

Authentication details must not be a part of the x-sap-ext-overview. Add authentication details in the security scheme section.

## x-servers

The Open API specification 2.0 does not support multiple hosts (and ports), neither are path templating or patterns supported. Some APIs need support for both multiple hosts and path templating in the host parameter. This is because the host and landscape vary between regions.

These features is supported in the `servers` property in the Open API specification v3.0. However, in Open API specification v2.0, the required configuration values can be added via the custom extension `x-servers`.

For more information about how to specify sandbox url and multiple hosts or production servers in OpenAPI 3.0, see the [Sandbox and Configure Information](#) sections in

See [here](#) to know more information about the differences between OpenAPI 2.0 and Open API 3.0 specifications

If values are supplied for host, schemes, and basepath, then they together form the root URL of the API. The root URL points to a Sandbox system or a test system wherein the API can be tested.

### ↔ Sample Code

```
{
 "host": "sandbox.api.sap.com",
 "schemes": [
 "https"
],
 "basePath": "/sap/v1"
}
```

For more information on how to define schemes, host and basepath in the API specification, see [API Designer \[page 490\]](#).

`x-servers` attribute is used when you want to specify multiple hosts, for example, to specify values for different servers located across various geographical boundaries. The example below shows how multiple hosts with path templates can be defined using `x-servers` attribute.

```
{
 "info": {
 "title": "This sample is a reference service, which runs on SAP BTP showcasing the e-commerce APIs for products, and suppliers.",
 "version": "1.0"
 },
 "x-servers": [
 {
 "url": "https://{appname}{accountname}.{landscapehost}/espm-cloud-web/espm.svc/secure",
 "description": "ESPM OData endpoints",
 "templates": {
```

```

 "appname": {
 "default": "espm",
 "description": "The application name used while deploying the ESPM
application"
 },
 "accountname": {
 "description": "The SAP BTP subaccount id where the application is
deployed"
 },
 "landscapehost": {
 "enum": [
 "hana.ondemand.com",
 "us1.hana.ondemand.com",
 "us2.hana.ondemand.com",
 "apl.hana.ondemand.com",
 "hanatrial.ondemand.com"
],
 "default": "hana.ondemand.com",
 "description": "The region(host) where the application is deployed."
 }
 }
}
]
}

```

## x-sap-direction

You can use this optional attribute to indicate the direction in which an API operates.

The value for this field would be one of the following:

- `inbound` (default)
- `outbound`
- `mixed` (both inbound and outbound)

### Sample Code

```

{
 "x-sap-direction": "outbound"
}

```

## x-sap-extensible

This optional attribute lets you indicate that a particular API is extensible.

The value for this field would be one of the following:

- `Manual`: API can be extended manually by custom fields in some business contexts.
- `Automatic`: API can be extended automatically using tools by custom fields in some business contexts.
- `No`: API can't be extended.

### Sample Code

```
{
 "x-sap-extensible": {
 "supported": "manual",
 "description": "API can be extended by custom fields on the following
business contexts (field usage for this API needs to be selected):\r\n*
Procurement: Purchasing Document (MM_PURDOC_HEADER)\r\n* Procurement:
Purchasing Document Item (MM_PURDOC_ITEM)\r\n\r\n[How to add an extension
field to an API](https://help.sap.com/viewer/9a281eac983f4f688d0deedc96b3c61c/
latest/en-US/57909455bf7c4fdd8bcf48d76c1eae33.html)"
 },
}
```

## x-targetEndpoint

For an API artifact to be working, you need to specify the target endpoint.

### Sample Code

```
{
 "info": {
 "title": "SAP Workflow service API",
 "description": "Provides functionality to work with SAP Workflow service.
You can, for example, start new workflow instances and work with tasks."
 },
 "x-targetEndpoint": "
https://sap.com"
}
```

## Handcrafting your Open API Specification

Handcrafting your Open API Specification to create a valid API artifact.

For an API artifact to be working, you need to have a valid base path, virtual host, and target endpoint.

For example,

### Sample Code

```
{
 "info": {
 "title": "SAP Workflow service API",
 "description": "Provides functionality to work with SAP Workflow service.
You can, for example, start new workflow instances and work with tasks."
 },
 "x-targetEndpoint": "
https://sap.com"
}
```

Virtual Host and Base Path information can be indicated using different attributes depending on the OpenAPISpecification version.

Here's an example for OpenAPI Specification 3.0:

#### Sample Code

```
{
 "servers": [
 {
 "url": "
https://vh1.com/apipath1"
 }
]
}
```

Where vh1.com indicates the virtual host and "apipath1" is the base path.

Here's an example for Swagger 2.0:

#### Sample Code

```
{
 "host": "https://vh1.com:443"
 "basePath": "apipath1"
}
```

Where vh1.com indicates the VH and "apipath1" is the base path.

## 1.5.2.2.2.5 Perform Additional Tasks in API Designer

Besides creating new APIs or editing existing APIs in the API designer, you can also do the following:

- To model Open API JSON content in the designer, choose **Paste > JSON**.
- To model an Open ODATA API, choose **Paste > OData Metadata**.
- To convert an API written in RAML to Open API, choose **Paste > RAML**.
- To import a YAML or JSON format file from your local server, choose **Import**.
- To download the API swagger specifications, choose **Download** and select JSON or YAML format.
- You can generate a server stub from the API definition file. The generated server stub can be used for deploying applications locally and as well as on Cloud Foundry.
  - From the **Generate Server Stub** dropdown menu, choose the required language in which you want to generate the server stub.
  - In the **Project Metadata** dialog window, enter the following information:
    - **Package Name:** The name of the package.
    - **GroupId:** The ID of the project's group.
    - **Artifact:** The ID of the artifact (project).
    - **Artifact Version:** The version of the artifact under the specified group.
  - Choose **Generate Project**.
  - The server stub is downloaded in a ZIP file. The generated server code contains stub methods and a README.md file with all the information required for building applications. If you have generated

a server stub for Cloud Foundry deployment, then the README.md file contains instructions for deploying application on Cloud Foundry. Each language creates a different README file. Go through it to learn about how to build and deploy the application.

- Choose [Save](#) to save the modeled API. You can choose to save the modeled API with a version by choosing the option [Save as Version](#).

### 1.5.2.3 Export and Import of API Definition

You can export an API definition along with their dependencies from the source and import the same to the target system.

#### Prerequisite

The role collection [APIPortal.Administrator](#) should be assigned to you.

During export a zip bundle gets generated, you can use this zip bundle to import the API definition in the target system.

#### Note

During the export and import of an API definition, only the standalone API definition and its associated entities like target endpoint, proxy endpoint, and resources get exported/imported. Other dependencies like API providers, KVMs, and certificates do not get imported/exported with the API definition; you must create them manually in the target system if they don't already exist in the target.

Import and export of any other entities like products, applications, API providers, KVMs, and certificates are not supported.

To export and import multiple API definitions and their dependencies, refer the following:

[Export an API Definition \[page 533\]](#)

Once you create an API in the , you can choose to export it.

[Import an API Definition \[page 534\]](#)

This topic describes how to import an existing API definition into the .

### 1.5.2.3.1 Export an API Definition

Once you create an API in the , you can choose to export it.

#### Prerequisites

The role collection [APIPortal.Administrator](#) should be assigned to you.

You have created an API in the with:

- Relevant resources and documentation.
- Policies attached to the API.

## Context

To export an API proxy, proceed as follows:

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. On the *APIs* tab page, choose the **Action** icon against the required API and then select the *Export* option.
4. Alternatively, you can open the required API, choose the breadcrumb and then select the option *Export*.
5. A .zip file is exported with contents as described in the [API Proxy Structure \[page 469\]](#). All the content related to the API documentation is available in Swagger\_json file. **The current state of the API, will also be available in the exported zip.**

### Note

The API state and related metadata are not exported.

## Related Information

[Import an API Definition \[page 534\]](#)

### 1.5.2.3.2 Import an API Definition

This topic describes how to import an existing API definition into the .

## Prerequisites

The role collection *APIPortal.Administrator* should be assigned to you.

The API proxy content is available as a .zip file and swagger json file. The contents adhere to the API proxy structure as defined in [API Proxy Structure \[page 469\]](#).

## 📌 Note

- If your API proxy uses an API provider that belongs to the type Cloud Integration flow, import of such an API proxy is currently not supported.
- Ensure that the API proxy name in the `<APIProxyName>.xml` file and the value of the `base_path` field (inside the `APIProxyEndpoint` file) is unique.
- `APIResources`, `DocumentationFileResource`, and `Policy` folders are not required in the .zip file.
- Ensure that the resource documentation is available in a single `OAS_json` file. Note that you cannot refer to external links in the API definitions within this json file.
- API Management supports import of API definitions in both OAS 2.0 and OAS 3.0. To know more about OAS 3.0 support in API Management, see [OpenAPI Specification 3.0 \[page 489\]](#).
- Optional: If you want to mention the API State of the API proxy you are importing, you need to update the following in the API proxy XML:
  - **API State:** The state of the API proxy. To know more about API States, see [API Proxy States \[page 580\]](#)
  - **Successor API:** If the API state of the API proxy you are importing is deprecated or decommissioned, you need to provide information about a successor API that the consumer should use.
  - **Release Metadata:** If the API state of the API proxy you are importing is deprecated or decommissioned, use this field to provide information about the following:
    - Reason for deprecation or decommissioning
    - Date of deprecation or decommissioning
    - External successor API: If you provide an external successor API, you will not use the Successor API parameter.
    - Changed By
    - Changed At

## 📄 Sample Code

```
<APIState>Deprecated</APIState>
<successorAPI>XYZ</successorAPI>
<releaseMetadata>{"reason": "undefined", "date": 1603132200000, "external
SuccessorAPI": "", "changed_at": 1602578356857, "changed_by": "xxx@abc.com
"}</releaseMetadata>
```

**File Resource** is a script or code snippet that can be attached to Flows using policies. An API proxy container supports definition of a number of Java, Python or XSL scripts. These scripts can be executed in the context of either a Java Script, Python Script or XSL Transformation policy. Once a Script is defined, it can be applied as either a Java Script policy, Python Script policy or XSL Transform policy in different Flows.

## Context

API Management provides the option to import an API definition.

## 📌 Note

When an API proxy is transported or exported individually or as a part of a product, by default, it gets imported to the target in the deployed state.

### Note

The API proxy zip can be successfully imported only if the providers associated with the API proxy in the source system are also present in the target system.

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. On the *APIs* tab page, choose **Create > Import API**.
4. In the *Import API* window:
  - a. Choose *Browse*. Navigate and choose the required API from your local file system.

You can attach files of type .zip and .json. When you import the API, you can create a new API or replace the existing API across landscapes seamlessly.

You can include the configurations for health monitor and load balancer in the .zip file. For more information, see [Load Balancing Across API Providers \[page 615\]](#).
  - b. You can choose to import either an API whose lifecycle will be managed by API Management, or an externally managed API, when importing the json file. Choose *Manage this API* to import an API whose lifecycle is to be handled by API Management.
  - c. Select a virtual host alias from the *Virtual Host* drop down.
  - d. If you want to version your API when uploading the .json file, select *Yes* for the *Create a Version* toggle button (this is set to *No* by default), and enter the version in the *Version* field that appears. The version will be appended to the name and prepended to the base path of the API, to create a unique API proxy version. For example, If the version you enter is v1, the name will be Name\_v1, and the basepath will be /v1/SalesOrder. To know more about versioning your API, see [API Versioning \[page 573\]](#).
  - e. If you want to list an externally managed API, when uploading the json file, choose *List as Externally Managed API*. This API will only be listed in , and it's lifecycle will not be managed. For more information, see [Externally Managed APIs \[page 588\]](#).
5. Choose *OK*.

### Note

The API state remains unchanged during import.

## Related Information

[Export an API Definition \[page 533\]](#)



## 1.5.3 Additional Configurations

To accelerate the connectivity to different backends, discover services, and expose them as managed APIs, you need to configure API providers. You use an API provider to define not only the details of the host you want an application to reach, but also to define any further details that are necessary to establish the connection, for example, proxy settings. For more information, see [API Providers \[page 537\]](#).

If you want to store data for retrieval at runtime, nonexpiring data that shouldn't be hard-coded in your API proxy logic. You can configure key value maps for retrieval of such data. Key value maps are a custom collection of encrypted key. For more information, see [Key Value Map \[page 554\]](#).

Certificates give you the credentials and the knowledge needed to create an API ecosystem with the scalability and security. Certificates are configured to confirm the identity of the caller and only if you recognize the identity the API is processed and data is returned. For more information, see [Manage Certificates \[page 558\]](#).

**Parent topic:** [Build API Proxies \[page 195\]](#)

### Related Information

[Key Components of an API \[page 196\]](#)

[Different Methods of Creating an API Proxy \[page 477\]](#)

### 1.5.3.1 API Providers

An API provider defines the connection details for services running on specific hosts whose details you want to access.

Use an API provider to define not only the details of the host you want an application to reach, but also to define any further details that are necessary to establish the connection, for example, proxy settings. You can configure connections to OData-hosted systems from .

#### API Providers Connecting to Backend System

If you want to configure the , solution to access data from a server that offers a specific service, for example, an SAP Gateway service, SAP HANA, SAP Process Integration/Process Orchestration, SAP S/4 HANA, or any 3rd party cloud solutions, you can manifest and expose the connection parameters as an API provider.

## Advantages of Creating API Providers

- You can connect to different backend on premise/cloud system.
- Discover services/interfaces.
- Simplifies on-premise connectivity.
- Simplifies configuration if the backend system changes.

## Related Information

[Create an API Provider \[page 538\]](#)

### 1.5.3.1.1 Create an API Provider

Define the details of the host you want an application to reach by creating an API provider.

## Prerequisites

- You're assigned the `APIPortal.Administrator` role.
- For *Open Connectors* type provider, fetch the *Organization Secret* and *User Secret* from the Open Connectors page.

### Note

Open Connectors is now a part of SAP Integration Suite. To activate the capability, choose *Add Capabilities* and choose *Extend Non-SAP Connectivity* in the *Activate Capabilities* screen. For more information, see *Activating and Managing Capabilities* and *Configuring User Access*. Once the capability is activated, navigate back to the home page and choose the *Edit* icon on the *Extend Non-SAP Connectivity* tile. On the *Integration Suite Open Connectors* page, choose the *Settings* icon at the bottom left of the screen and note down the *Organization Secret* and *User Secret*.

You're navigated to *Integration Suite* homepage. At first, you have to activate the Open Connectors capability.

- The following prerequisites are applicable for *On-Premise* type provider only.
  1. Expose the on-premise system using *Cloud Connector*. For more details, see [here](#).
  2. From your *Subaccount*, choose *Cloud Connectors* from the left-hand pane and validate if the system exposed in the previous step is visible in the list.

## Context

If you want to configure the API Management solution to access data from a server that offers a specific service, for example, SAP Gateway service, SAP HANA, SAP Process Integration/Process Orchestration, SAP S/4HANA, or any 3rd party cloud solutions, we recommend configuring the solution as an API provider in the connection parameters. This enables you to configure connections to OData-hosted systems.

### Note

You can see the following tutorial for visual instructions on how to [Create an API Provider System](#).

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose ► *Configure* ► *APIs* ▾.

The lists of API providers appear.

3. Choose *Create*.
4. Enter a name and a description for the API provider.

### Note

Do not use the names mentioned in following list:

- DEST\_CI
- APIMGMT\_PORTAL
- ContentCatalog
- ON\_PREM
- Apimgmt\_rt
- Apimgmt\_rt\*
- TC
- TC\_\*

5. Go to the *CONNECTION* section, enter the required details based on the type of connection you selected.
  - Choose *Internet* to connect to a system on Cloud.
  - Choose *On Premise* to connect to the on-premises system through Cloud Connector.
  - Choose *Open Connectors* to connect third-party APIs via harmonized RESTful APIs.
  - Choose *Cloud Integration* to access all the service endpoints.
  - *Internet* connection type:

Internet

Field Name	Description
Host	Enter a value for the host in the format "example.com".
Port	Enter 443 as the port number for SSL and 8080 for all other ports.
Use SSL	Select the checkbox to secure the connection using SSL (HTTPS protocol).
	<div><p><b>Note</b></p><p>Setting this option is sufficient to set up the SSL connection. If you want to use SSL to secure connections to the configured destination, then upload the server certificate in the SAP BTP cockpit.</p></div>
Trust store certificate	By default, API Management trusts all TLS certificates. To validate a certificate, create a truststore to configure API Management to validate the certificate. Specify the trust store certificate details in the API provider.
Key store certificate	Specify the key store certificate details.

- *On Premise* connection type:

Field Name	Description
Host	Enter a value for the host in the format "example.com". Ensure that you provide the cloud connector virtual host.
Port	Enter the port number exposed in the <i>Cloud Connector</i> as per the host.
Location ID	If you have configured multiple cloud connectors to your SAP BTP account, then provide the location ID of the cloud connector that you want to add.
	<div><p><b>Note</b></p><p>The <i>Location ID</i> value for a cloud connector is set while configuring the cloud connector.</p></div>
Authentication	Choose an authentication method: <ul style="list-style-type: none"><li>• <i>None</i>: No Authentication. This is a pass through flow and validation is initiated at the layer.</li><li>• <i>Principal Propagation</i>: Enables authentication of a message in the receiver system with the same user that issued the message in the corresponding sender system. For more details, see <a href="#">Principal Propagation [page 546]</a>.</li></ul>

Field Name	Description
Additional Property	Select <i>sap-client</i> from the dropdown and enter the three-digit client ID. Client-specific data is identified with the client identifier. SAP-client configured in API provider has the highest precedence for runtime calls, metadata fetch, and discovery.

**Note**

The metadata is fetched from the specified client ID.

**Note**

Adding trust-store and key-store certificates isn't supported for the on-premise system.

- *Open Connectors* connection type:

Open Connectors

Field Name	Description
Region	Select a region from the dropdown list.
Host	Value is auto populated based on the selected region.
Port	Value is auto populated.
Organization Secret	Enter the organization secret. For more information on how to obtain the value, see prerequisites section.
User Secret	Enter the user secret. For more information on how to obtain the value, see prerequisites section.

- *Cloud Integration* connection type:

SAP Cloud Integration

Field Name	Description
Cloud Integration Host	<p>Enter a tenant-specific address of the Cloud Integration system which can be accessed to by a dialog user (for example, an integration developer using the Web UI) or through the OData API.</p> <p>Address of Web UI: https://&lt;Cloud Integration Host&gt;/itspaces.</p> <p>Address of Cloud Integration OData API: https://&lt;Cloud Integration Host&gt;/api/v1.</p>
Port	443 (Default value)

Field Name	Description
Authentication	<p>The authentication type can be:</p> <ul style="list-style-type: none"> <li>• <a href="#">Basic</a> for connecting requests to server. This Basic authentication requires you to enter a username and password.</li> <li>• <a href="#">OAuth2ClientCredentials</a> for connecting requests to the server. The OAuth2ClientCredentials authentication needs you to provide the following details: <ul style="list-style-type: none"> <li>• Client ID</li> <li>• Client Secret</li> <li>• Token URL</li> </ul> </li> </ul> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p><b>Note</b></p> <p>The values you provide for the above, are the ones associated with your Cloud Integration tenant.</p> </div>

Also see:

- [Setting Up OAuth for Cloud Integration in Cloud Foundry \[page 544\]](#)

6. In the *CATALOG SERVICE SETTINGS* section, enter the required details.

Catalog Service Settings

Field Name	Description
Path Prefix	Specify the path prefix.
Service Collection URL	Specify the relative path from where the catalog service should be fetched. For example, <code>/CATALOGSERVICE/ServiceCollection</code> .
	<div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p><b>Note</b></p> <p>The option is only relevant for SAP Gateway-enabled SAP NetWeaver systems. A catalog service retrieves a list of all available OData services on SAP Gateway. It's based on a catalog service pattern proposed by Microsoft and consists of an implementation approach of the catalog service pattern in the context of SAP Gateway. Only use the field if you want to fetch services from SAP Gateway. For more information on the catalog service, see <a href="#">here</a>.</p> </div>
Catalog URL	Automatically populates the complete URL in the format <code>https:&lt;host&gt;:&lt;port&gt;/&lt;path prefix&gt;/Catalog Service</code> .

Field Name	Description
Authentication	Select the authentication method to be used for connection requests to the server. By default, the authentication type is set to <i>None</i> . If you choose <i>Basic</i> , then provide a user name and password as authentication credentials in the respective fields.

### Note

Catalog service settings aren't applicable for Open Connectors type connection and SAP Cloud Integration.

### 7. Save the changes.

Choose [Configure](#) > [API Provider](#) to view the newly created API provider. Further, you can test the connection of the API provider, by navigating to the API provider's details page and selecting [Test Connection](#).

### Note

The following table lists the attributes considered for testing the connection:

Test Connection

Type	Supported Attributes	Unsupported Attributes
Internet	<ul style="list-style-type: none"> <li><a href="#">Connection</a> attributes <ul style="list-style-type: none"> <li>Host</li> <li>Port</li> <li>UseSSL</li> </ul> </li> <li><a href="#">Catalog Service Settings</a> attributes <ul style="list-style-type: none"> <li>Path Prefix (if available)</li> <li>Service Collection URL (if available)</li> <li>Trust All</li> <li>Authentication Type</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Connection</a> attributes <ul style="list-style-type: none"> <li>Trust Store</li> <li>Key Store Certificate</li> </ul> </li> </ul>

A fixed socket timeout of 5 seconds and connection timeout of 3 seconds is added to the http client, which makes the HEAD call to the API provider.

## 1.5.3.1.1.1 Setting Up OAuth for Cloud Integration in Cloud Foundry

For Cloud Integration type Provider there are two types of authentications: Basic and OAuth2ClientCredentials.

### Context

Use Basic authentication for sending requests to server. Basic authentication requires you to enter a username and password.

Use OAuth2ClientCredentials for connecting requests to the server. This OAuth2ClientCredentials authentication requires you to enter *clientId*, *clientSecret*, and *tokenUrl*.

#### Note

The values you provide for the above, are the ones associated with your Cloud Integration tenant.

**Generate *clientId*, *clientSecret*, and *tokenUrl* Associated with Your Cloud Integration Tenant**

### Procedure

1. In your web browser, open the *SAP BTP Cockpit* - <https://eu-access.cockpit.btp.cloud.sap>.
2. From your *Subaccount*, navigate to *Spaces* in your Cloud Foundry environment and choose **Services** **Service Marketplace**.
3. Choose *Process Integration Runtime* and choose *Create*.
4. In the *Create Instance* dialog that opens, choose the *api* plan.
5. In the section *Specify parameters*, paste the following JSON codes, to assign a specific role.

```
"roles": [
 "AuthGroup_IntegrationDeveloper" ,
 "AuthGroup_Administrator"]
```

6. Choose *Next* until you reach the *Confirm* section
7. In the *Confirm* section, enter a unique *Instance Name* and choose *Finish*.

### Results

The creation of service instance is successful.



## Next Steps

Now, for the created service instance, generate a service key from the steps given below:

## Creating a Service Key

### Procedure

1. Choose the created service instance link from the visible list.
2. In the left-hand pane, navigate to ► [Service Keys](#) ► [Create Service Key](#) 🗒.
3. In the [Create Service Key](#) dialog that opens, provide a [Name](#) and [Description](#) (optional).
4. Choose [Save](#). The client credentials like [url](#), [clientId](#), [clientSecret](#), and [tokenUrl](#) details appear for the given instance name.

The [clientId](#) and [clientSecret](#) are necessary credentials required to fetch the Bearer Token.

The [tokenUrl](#) is used to fetch the Bearer Token.

```
Example:
 "clientId": "sb-37b5d80e-27f9-4b73-bf58-c29f2a156df8!b16289|it!
b12456",
 "clientsecret": "311ccbf6-86ef-48bb-910c-
a7eee79ea29f$h6xDUyIUoe45qiDGGBhzK9zvDczFeePUHDNLBshnMCk=",
 "url": "https://isuitetestcpi.it-
adev001.cfapps.sap.hana.ondemand.com",
 "tokenurl": "https://
isuitetestcpi.authentication.sap.hana.ondemand.com/oauth/token"
```

[Copy](#) the client credentials in a notepad and use it while creating the API Provider.

## To Discover and Create an API Proxy for Cloud Integration

### Context

The service keys provide you with [clientId](#) and [clientSecret](#). The [clientId](#) can be used as username and secret can be used as password if you would like to connect to your integration flows of Cloud Integration via Basic Authentication. Alternatively, you can leverage the [clientId](#), [clientSecret](#), and [tokenUrl](#) from the service keys file to get the OAuth access token and then connect to your integration flow of Cloud Integration via OAuth access token approach. For more information, see [Creating an API Proxy using SAP Cloud Integration API Provider \[page 603\]](#).

## 1.5.3.1.1.2 Principal Propagation

Principal propagation is a process that provides a secure way of forwarding the identity of a cloud user to the Cloud Connector, and from there to an on-premise system. The user information is kept confidential and, even more importantly, not changed during transit.

### Prerequisites

- Your on-premise back-end system supports X.509 certification.
- On-premise connectivity is enabled in your Cloud Connector application, Cloud Controller settings. To verify the same:
  - Navigate to ► [Cloud To On-Premise](#) ► [PRINCIPAL PROPAGATION](#) ► in your Cloud Connector application, Cloud Controller settings.
  - Using the Name and Description column, identify the IDP connected to your Cloud Foundry subaccount, which needs to be trusted for principal propagation.
  - In the *Trusted* column, check if on-premise connectivity is enabled, else select the checkbox to enable it.

For information on how to connect your system via Cloud Connector, see [Cloud Connector](#)

### Context

A principal forwarded to the API Proxy is validated with the IDP connected to the user's subaccount on Cloud Foundry where is enabled. To obtain an OAuth token, which is validated, the user has to create credentials using on-premise-connectivity plan in the Cloud Foundry environment.

In API Management the Principal Propagation supports two flows namely:

- **Principal Propagation from the Neo to the Cloud Foundry Environment:** Enable an application in your subaccount in the Neo environment to access an API Management proxy created on a Cloud Foundry based API Management without a user login. For this scenario to work, the Neo subaccount needs to be trusted by the Cloud Foundry subaccount where API Management, API portal is enabled. Now, the application on Neo can call API Management proxy using OAuth2SAMLBearer destination. For step-by-step details, see [Principal Propagation from the Neo to the Cloud Foundry Environment \[page 547\]](#).
- **Principal Propagation from the same Cloud Foundry subaccount:** Enable an application in your subaccount in the Cloud Foundry environment to access an API Management proxy created on the same Cloud Foundry based API Management without a user login. The JWT user token in your application can be exchanged with the API Management token using the Service Key credentials created for API Management. The application on Cloud Foundry can call API Management proxy using OAuth2UserTokenExchange destination. For step-by-step details, see [Principal Propagation from the Same Cloud Foundry Subaccount \[page 550\]](#).

## Related Information

[Accessing On-Premise Systems through API Management \[page 150\]](#)

### 1.5.3.1.1.2.1 Principal Propagation from the Neo to the Cloud Foundry Environment

#### Prerequisites

- You have created a Service Key by creating a service instance using the on-premise connectivity plan. For more details, see [Accessing On-Premise Systems through API Management \[page 150\]](#).

#### Context

Enable an application in your subaccount in the Neo environment to access an API Management proxy created on a Cloud Foundry based API Management without a user login. For this scenario to work, the Neo subaccount needs to be trusted by the Cloud Foundry subaccount where API Management is enabled. Now, the application on Neo can call API Management proxy using OAuth2SAMLBearer destination.

#### Procedure

- Create Trust between the Neo Subaccount and the Cloud Foundry Subaccount. For detailed steps, see [Create Trust between Subaccounts](#).
- Create a Destination to the API Proxy that you want to call using principal propagation.

New Destination

Field	Description
Name	Technical name of the destination. It can be used later on to get an instance of that destination. It must be unique for the global account.
URL	Enter the API Proxy URL for the proxy you want to call.
Authentication	OAuth2SAMLBearerAssertion
Proxy Type	Internet

Field	Description
Audience	<p>Copy the value of entityID property of the SAML 2.0 metadata representing your subaccount in the Cloud Foundry environment.</p> <div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p>→ Tip</p> <p>You can open the metadata of the subaccount in the Cloud Foundry environment using the tokenUrl you obtain from the Service Key:</p> <p>https://&lt;your subaccount's subdomain&gt;.authentication.&lt;SAP BTP host&gt;/saml/metadata</p> <p>Example:</p> <p>https://&lt;tenant-specific-route-for-your-business-app&gt;.sap.hana.ondemand.com/oauth/token</p> </div> <p>Example of audience/entityID:</p> <p>demo.aws-live-us10</p>
Client Key	Enter the <i>clientId</i> obtained from the <i>Service Key</i> in the created service instance using on-premise connectivity plan.
Token Service URL	<p>Enter the token url obtained from the <i>Service Key</i> in the created service instance using on-premise connectivity plan.</p> <p>The token service URL is defined in the Location attribute of the element marked as AssertionConsumerService, like this :</p> <pre>&lt;md:AssertionConsumerService Location="&lt;Token Service URL&gt;" Binding="urn:oasis:names:tc:SAML:2.0:bindings:URI" index="1"/&gt;</pre> <div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p>→ Tip</p> <p>You can open the metadata of the subaccount in the Cloud Foundry environment using the tokenUrl you obtain from the Service Key:</p> <p>Example: https://&lt;tenant-specific-route-for-your-business-app&gt;.sap.hana.ondemand.com/oauth/token</p> </div>
Token Service Password	Enter the <i>clientSecret</i> obtained from the <i>Service Key</i> in the created service instance using on-premise connectivity plan.
System User	Empty

If you have not generated the client credentials (clientId, ClientSecret, tokenUrl and application url) yet, see [Accessing On-Premise Systems through API Management \[page 150\]](#)

For more information on creating a destination, see [here](#).

- Use the following sample source code to consume the above created destination in your application.

### Sample Code

```
protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
 // TODO Auto-generated method stub
 try {
 // Look up the connectivity configuration API
 Context ctx = new InitialContext();
 ConnectivityConfiguration configuration =
(Configuration) ctx
 .lookup("java:comp/env/connectivityConfiguration");
 // Get destination configuration
 DestinationConfiguration destConfiguration =
configuration.getConfiguration("APIProxyTest"); //Name of Destination
 if (destConfiguration == null) {
 response.sendError(HttpServletResponse.SC_INTERNAL_SERVER_ERROR,
String.format(
 "Destination %s is not found. Hint:" +
 " Make sure to have the destination
configured.",
 "pptest"));
 return;
 }

 AuthenticationHeaderProvider authHeaderProvider =
(AuthenticationHeaderProvider) ctx
 .lookup("java:comp/env/myAuthHeaderProvider");
 // retrieve the authorization header for OAuth SAML Bearer principal
propagation
 List<AuthenticationHeader> samlBearerHeader = authHeaderProvider
 .getOAuth2SAMLBearerAssertionHeaders(destConfiguration);
 LOGGER.debug("JWT token from CF XSUAA: " +
samlBearerHeader.get(1).getValue());
 response.getWriter().println("JWT token from CF XSUAA: " +
samlBearerHeader.get(1).getValue());
 String destinationURL = destConfiguration.getProperty("URL");
 URL url = new URL(destinationURL);
 HttpURLConnection connection = (HttpURLConnection)
url.openConnection();
 connection.setRequestMethod("GET");
 connection.setRequestProperty("Authorization",
samlBearerHeader.get(1).getValue());
 connection.setConnectTimeout(10000);
 connection.setReadTimeout(60000);
 int responseCode = connection.getResponseCode();
 BufferedReader in = new BufferedReader(
 new InputStreamReader(connection.getInputStream()));
 String inputLine;
 StringBuffer responseBody = new StringBuffer();
 while ((inputLine = in.readLine()) != null) {
 responseBody.append(inputLine);
 }
 connection.disconnect();
 response.getWriter().println("Response Status : " + responseCode);
 response.getWriter().println("Response Body" +
responseBody.toString());

 response.getWriter().close();
 } catch (Exception e) {
 // Connectivity operation failed
 String errorMessage = e.getMessage();
 LOGGER.error("Connectivity operation failed", e);
 }
}
```

```
 response.sendError(HttpServletResponse.SC_INTERNAL_SERVER_ERROR,
 errorMessage);
 }
}
```

Principal propagation from a Neo subaccount to Cloud Foundry Subaccount is established.

## 1.5.3.1.1.2.2 Principal Propagation from the Same Cloud Foundry Subaccount

Transfer the principal (user details) within the same Cloud Foundry subaccount by exchanging tokens.

### Prerequisites

- You have created a [Service Key](#) by creating a service instance using the on-premise connectivity plan. For more details, see [Accessing On-Premise Systems through API Management \[page 150\]](#).

### Context

When an application has to communicate with another application in the same subaccount on Cloud Foundry, exchange of token has to take place for secure passage of user details. To make this step easier, you can use the OAuth2UserTokenExchange authentication type, where the Destination service performs all these steps automatically and lets you simplify your application development. Therefore, the JWT user token in your application can be exchanged with the API Management token using the [Service Key](#) or client credentials by using OAuth2UserTokenExchange. For more details on OAuth2UserExchange destination, see [here](#)

### Procedure

- Create a Destination to the API Proxy that you want to call using principal propagation. Enter the following details while configuring a destination of type OAuth2UserTokenExchange. Also, if you have not generated the client credentials (clientId, ClientSecret, tokenUrl and application url) yet, see [Accessing On-Premise Systems through API Management \[page 150\]](#).

#### Sample Code

```
Example:
Type=HTTP
clientId=Client id from the credentials
Authentication=OAuth2UserTokenExchange
Name= Name of the destination
tokenServiceURL= Token url from the credentials
ProxyType=Internet
URL=API Proxy URL for the proxy to be called
tokenServiceURLType=Dedicated
clientSecret=<Client Secret from the credentials>
```

- Exchange the JWT user token in the application with the [API Management, API, portal](#) application via OAuth2UserTokenExchange authentication type for HTTP destinations. For more detailed information, see [here](#).

### Next Step:

Use the token received from the OAuth2UserTokenExchange to call an API Proxy and consume the above -created destination from your application.

## 1.5.3.1.1.2.3 Principal Propagation Between Two Different Subaccounts in Cloud Foundry Environment

Propagate the identity of a user between two Cloud Foundry applications that are located in different subaccounts or regions.

### Prerequisites

- You have two applications (application 1 and application 2) deployed in Cloud Foundry environment in different subaccounts in the same region or in different regions. Application 1 resides in subaccount 1 and application 2 resides in subaccount 2.
- You have an instance of the Destination service bound to application 1.
- You have a user JWT (JSON Web Token) in application 1 where the call to application 2 is performed.

### Procedure

#### Assemble IdP Metadata for Subaccount 1

1. From [Subaccount 1](#) > [Destinations](#) > [Download Trust](#) download the X.509 certificate of subaccount 1. The content of the file is shown as:

```
-----BEGIN CERTIFICATE-----<content>-----END CERTIFICATE-----
```

We refer to the value of <content> as `${S1_CERTIFICATE}`.

2. In the cockpit, navigate to the overview page of subaccount 1. Here you can see the landscape domain, subaccount ID, and subdomain. Below we refer to the landscape domain as `${S1_LANDSCAPE_DOMAIN}`, to the subaccount ID as `${S1_SUBACCOUNT_ID}` and to the subdomain as `${S1_SUBDOMAIN}`.
3. In your browser, call `https://${S1_SUBDOMAIN}.authentication.${S1_LANDSCAPE_DOMAIN}/saml/idp/metadata` and download the XML file. Within the XML file you can find the following structure:

#### Sample Code

```
<?xml version="1.0" encoding="UTF-8"?>
<md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
ID="<id>" entityID="<alias>">
 ...
</md:EntityDescriptor>
```

We refer to the value of <alias> as `${S1_ALIAS}`.

4. Assemble the new IdP metadata for subaccount 1 by replacing the `${...}` placeholders in the following template with the values determined in the previous steps:

#### Sample Code

```
<ns3:EntityDescriptor
ID="cfapps.${S1_LANDSCAPE_HOST}/${S1_SUBACCOUNT_ID}"
```

```

entityID="cfapps.${S1_LANDSCAPE_HOST}/${S1_SUBACCOUNT_ID}"
xmlns="http://www.w3.org/2000/09/xmldsig#"
xmlns:ns2="http://www.w3.org/2001/04/xmlenc#"
xmlns:ns4="urn:oasis:names:tc:SAML:2.0:assertion"
xmlns:ns3="urn:oasis:names:tc:SAML:2.0:metadata">
<ns3:SPSSODescriptor AuthnRequestsSigned="true"
 protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
 <ns3:KeyDescriptor use="signing">
 <KeyInfo>
 <KeyName>${S1_ALIAS}</KeyName>
 <X509Data>
 <X509Certificate>
 ${S1_CERTIFICATE}
 </X509Certificate>
 </X509Data>
 </KeyInfo>
 </ns3:KeyDescriptor>
</ns3:SPSSODescriptor>
<ns3:IDPSSODescriptor
 WantAuthnRequestsSigned="true"
 protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
 <ns3:KeyDescriptor use="signing">
 <KeyInfo>
 <KeyName>${S1_ALIAS}</KeyName>
 <X509Data>
 <X509Certificate>
 ${S1_CERTIFICATE}
 </X509Certificate>
 </X509Data>
 </KeyInfo>
 </ns3:KeyDescriptor>
</ns3:IDPSSODescriptor>
</ns3:EntityDescriptor>

```

## Establish Trust between Subaccount 1 and Subaccount 2

1. In the cockpit, navigate to the overview page for subaccount 2.
2. From the left panel, select **Security** > **Trust Configuration**. Choose **New Trust Configuration**.
3. Paste the assembled IdP metadata for subaccount 1 in the **<Metadata>** text box and uncheck **Available for User Logon**.
4. Choose **Parse**.
5. Enter a **<Name>** for the trust configuration and choose **Save**.

### Note

Additionally, you must add users to this new trust configuration and assign appropriate scopes to them.

## Create an OAuthSAMLBearerAssertion Destination for Application 1

1. In the cockpit, navigate to the overview page for subaccount 2.
2. Here you can see the landscape domain, subaccount ID and subdomain of subaccount 2. We refer to the landscape domain as `${S2_LANDSCAPE_DOMAIN}`, to the subaccount ID as `${S2_SUBACCOUNT_ID}` and to the subdomain as `${S2_SUBDOMAIN}`.
3. In your browser, call `https://${S2_SUBDOMAIN}.authentication.${S2_LANDSCAPE_DOMAIN}/saml/idp/metadata` and download the XML file. Within the XML file, you can find the following structure.

### Sample Code

```
<?xml version="1.0" encoding="UTF-8"?>
```



```

<md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
ID="<id>" entityID="<alias>">
...
</md:EntityDescriptor>
<md:AssertionConsumerService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:URI"
Location="<token>" index="1"/></md:SPSSODescriptor>
</md:EntityDescriptor> Token= ${S2_Token_URL}

```

We refer to the value of <alias> as `${S2_ALIAS}`.

- In cockpit, navigate to subaccount 1.
- From the left panel, select **Connectivity** > **Destinations**.
- Choose *New Destination* and configure the values as described below. Replace the `${...}` placeholders with the values you determined in the previous steps and sections.

Property	Value
Name	Choose any name for your destination. You use this name to request the destination from the Destination service.
Type	<b>HTTP</b>
URL	The URL of application 2, identifying the resource you want to consume.
Proxy Type	<b>Internet</b>
Authentication	<b>OAuth2SAMLBearerAssertion</b>
Audience	<b>entityId</b>
Client Key	The <i>clientid</i> of the XSUAA instance in subaccount 2. Can be acquired via a binding or service key.
Token Service URL	The URL of the XSUAA instance in subaccount 2. Can be acquired from <token>.
Token Service URL Type	<b>Dedicated</b>
Token Service User	The <i>clientid</i> of the XSUAA instance in subaccount 2. Can be acquired via a binding or service key.
Token Service Password	The <i>clientsecret</i> of the XSUAA instance in subaccount 2. Can be acquired via a binding or service key.

#### Additional Properties

Property	Value
nameIdFormat	<b>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</b>
authnContextClassRef	<b>urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession</b>

- Choose *Save*.

## Consume the Destination and Execute the Scenario

To perform the scenario and execute the request from application 1, targeting application 2, proceed as follows:

1. Decide on where the user identity will be located when calling the Destination service. For details, see [User Propagation via SAML 2.0 Bearer Assertion Flow](#). This will determine how exactly you will perform step 2.
2. Execute a "find destination" request from application 1 to the Destination service. For details, see [Consuming the Destination Service](#) and the [REST API documentation](#).
3. From the Destination service response, extract the access token and URL, and construct your request to application 2. See "[Find Destination](#)" [Response Structure](#) for details on the structure of the response from the Destination service.

### 1.5.3.2 Key Value Map

A key value map lets you create and manage collections of arbitrary key value pairs for any number of API proxies. Each key value pair is stored in a map as an entry.

#### Note

It's recommended that you avoid making any concurrent inserts and updates to the same key value map (KVM) scoped to the environment level as it may cause loss of data.

As a work-around, use API proxy scoped key value map.

#### Caution

Don't use Key Value Maps to store your logs as this can impact API proxy runtime flow. Instead, use the message logging policy to write your logs to external endpoints.

#### [Create a Key Value Map \[page 555\]](#)

Create a key value map using the create and manage collections of arbitrary key value pairs for any number of API proxies.

#### [Update a Key Value Map \[page 556\]](#)

Update a key value map.

#### [Delete a Key Value Map \[page 557\]](#)

Delete a key value map which is not in use.

## 1.5.3.2.1 Create a Key Value Map

Create a key value map using the create and manage collections of arbitrary key value pairs for any number of API proxies.

### Prerequisites

You're assigned the admin role.

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. Choose *Key Value Maps* and choose *Create*.
4. On the *Create* screen, provide the following details:

Field Name	Description
Name	Key value map name
Encrypted	Select this checkbox if you want to encrypt the values.
Key	Key name
	<b>⚠ Restriction</b> You can't use // as a part of the Key name.
Value	Value for the key

5. Choose *Save*.

**Task overview:** [Key Value Map \[page 554\]](#)

### Related Information

[Update a Key Value Map \[page 556\]](#)

[Delete a Key Value Map \[page 557\]](#)

[Delete a Key Value Map \[page 557\]](#)

[Update a Key Value Map \[page 556\]](#)

## 1.5.3.2.2 Update a Key Value Map

Update a key value map.

### Prerequisites

- You are assigned the admin role.
- You have created a key value map.

### Context

You are updating a key value map to either add an entry, delete an entry, or update the value for an existing entry.

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose ► [Configure](#) ► [APIs](#) ▾.
3. Choose [Key Value Maps](#) and select the key value map that you want to edit.
4. Choose [Edit](#) to perform the following:
  - a. If you want to add an entry, choose [Add](#) and provide the [Key](#) and [Value](#).
  - b. If you want to delete an entry, choose the delete icon in the [Action](#) column. Save the changes.
  - c. If you want to update an entry, select the required entry and update the [Value](#) field.

#### ⓘ Note

You can only update the [Value](#) field and not the [Key](#) field.

5. Choose [Save](#).

You can view the updated key value map by navigating to ► [API Portal home page](#) ► [Configure](#) ► [Key Value Map](#) ▾.

**Task overview:** [Key Value Map \[page 554\]](#)

### Related Information

[Create a Key Value Map \[page 555\]](#)

[Delete a Key Value Map \[page 557\]](#)

[Create a Key Value Map \[page 555\]](#)

[Delete a Key Value Map \[page 557\]](#)

## 1.5.3.2.3 Delete a Key Value Map

Delete a key value map which is not in use.

### Prerequisites

You are assigned the admin role.

### Context

You are deleting a key value map.

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. Choose *Key Value Maps* and select the key value map that you want to delete.
4. In the *Actions* column, choose the delete icon for the key value map.

**Task overview:** [Key Value Map \[page 554\]](#)

### Related Information

[Create a Key Value Map \[page 555\]](#)

[Update a Key Value Map \[page 556\]](#)

[Create a Key Value Map \[page 555\]](#)

[Update a Key Value Map \[page 556\]](#)

## 1.5.3.3 Manage Certificates

By using certificates, you can ensure that whenever a call is made to your API, there's a certificate attached to it that confirms the identity of the caller and only if you recognize this identity the API can be processed and return data can be provided.

### Prerequisites

In this procedure, we assume that you're aware of the process of creating certificates using any tool of your own choice.

### Context

You need to create key store and trust store certificates to configure 2-way SSL (Secure Sockets Layer). SSL is the standard security technology for establishing an encrypted link between a web server and a web client, such as a browser or an app. An encrypted link ensures that all data passing between the server and the client remains private. To use SSL, a client makes a secure request to the server by using the encrypted https:// protocol, instead of the unencrypted http:// protocol. In , you can associate the certificates with the API Provider at the time of API provider registration. This process provides more secure way to access API provider.

Whenever the existing certificate expires, you can upload a new certificate and associate the same with the API provider. You can't upload an expired certificate.

The following are the supported file format for certificates: .cer, .jar (signed jar), .der, .pem, .p12, .pkcs

#### ⓘ Note

Whenever you're trying to establish a connection between your client and the API Management gateway, certificate pinning ensures that the TLS connection is set up using a particular certificate only. This can help you in situations where you may run into the risk of trusting certificate authorities that you shouldn't. However, the certificate pinning feature isn't supported currently in API Management.

### Procedure

1. Log on to .
2. Choose the navigation icon on the top-left and choose ► [Configure](#) ► [APIs](#) ► .
3. Select the [Certificates](#) tab.
4. Choose [Create](#).
5. Select the type of certificate that you want to create.
  - [Trust Store](#) - A truststore contains certificates used to verify certificates received as part of SSL handshaking. If the certificate received by an SSL client is signed by a valid certificate authority (CA),

then the client makes a request to the CA to authenticate the certificate else self-signed certificate can be uploaded in the truststore.

### Note

Since client certificate chains are used in the authentication process to establish the identity of clients accessing the API Management service, it is important to ensure that these chains have sufficient security measures in place. Weak client certificate chains lack the necessary security measures and are therefore vulnerable to attacks. As a result, weak client certificate chains have been deprecated. For more detailed information, please [3418201 - Deprecation of Weak Client Certificate Chains in API Management \(sap.corp\)](#).

- **Key Store** - A keystore contains an SSL certificate and private key used to validate the server during SSL handshaking.

The examples in this document show the SSL cert and key defined as PEM files, which comply with the X.509 format. If your cert or private key isn't defined by a PEM file, you can convert it to a PEM file by using utilities such as openssl. However, many .crt files and .key files are already in the PEM format. If these files are text files, and are enclosed in:

```
-----BEGIN CERTIFICATE-----
```

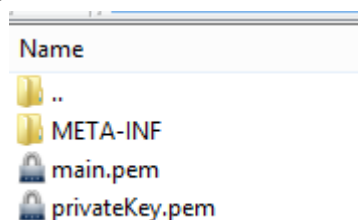
```
-----END CERTIFICATE-----
```

and:

```
-----BEGIN ENCRYPTED PRIVATE KEY-----
```

```
-----END ENCRYPTED PRIVATE KEY-----
```

6. You can either choose to use an existing store or create a new store and then add a new certificate in that store.
7. If you choose to create a new store, then enter the following details: store name, certificate name and appropriate description.
8. If you have chosen to create a key store, then execute the sub-steps below:
  - a. Create a JAR file containing your private key, certificate, and a manifest. For example, the JAR file must contain the following files and directories: /META-INF/descriptor.properties, <main>.pem,



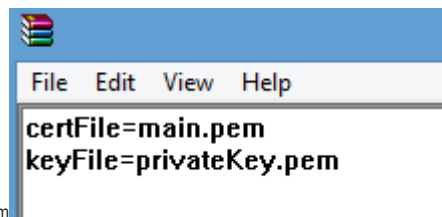
```
<privateKey>.pem
```

### Note

Ensure to create certificate with unique name. In case if a certificate with the same name exists in your system, then the newly created certificate will overwrite the existing one, and you'll lose your old certificate data.

A keystore JAR can contain only one certificate. If you have a certificate chain, all certs in the chain must be appended into a single PEM file, where the last certificate is signed by a CA. The certs must be appended to the PEM file in the correct order, meaning: cert -> intermediate cert(1) -> intermediate cert(2) -> ... -> root

- b. In the directory containing your key pair and certificate, create a directory called /META-INF. Then, create a file called descriptor.properties in /META-INF with the following contents:



```
certFile=<main>.pem keyFile=<privateKey>.pem
```

- c. Generate the JAR file containing your key pair and certificate: `$ jar -cf myKeystore.jar main.pem privateKey.pem`
  - d. Add descriptor.properties to your JAR file: `$ jar -uf myKeystore.jar META-INF/descriptor.properties`
  - e. Upload the JAR file.
9. If you have chosen to create a trust store, then upload only the PEM file.
  10. Choose [Create](#).

Once you create a certificate, you can then associate it with the API provider at the time of API provider registration.

### 1.5.3.4 Working with References

A reference is a variable that holds the name of the keystore or truststore, instead of directly specifying the name in the virtual host configuration.

The benefit of using a reference is that you can easily change the value of the reference to switch the keystore or truststore used by the virtual host. This is particularly useful when the current keystore or truststore's certificate is about to expire.

#### Note

References can only be used for the keystore and truststore, not for the certificates.

When changing the reference to a keystore, make sure that the alias name of the certificate remains the same as in the old keystore.

#### Note

When configuring a virtual host, you have the option to specify a keystore or truststore using a reference. For more information on how to configure custom domain virtual host and client certificate based authentication using references, see [Configuring Additional Virtual Host in Cloud Foundry Environment \[page 124\]](#).

You can create, update, fetch and delete any certificate store references via service calls. For more information, see the following:

#### [Creating the References \[page 561\]](#)

You can use the CertificateStoreReferences API to create a new certificate store reference.

#### [Updating the References \[page 563\]](#)

You can use the CertificateStoreReferences API to update a certificate store reference via service calls.



[Reading the References \[page 564\]](#)

Use the CertificateStoreReference API to get the list of certificate store references.

[Deleting the References \[page 566\]](#)

Use the CertificateStoreReference API to delete the certificate store reference via service calls.

## 1.5.3.4.1 Creating the References

You can use the CertificateStoreReferences API to create a new certificate store reference.

### Prerequisites

- You must have a service key for the **APIPortal.Administrator** role.  
To know more about creating a service key for accessing APIs in the API portal, see the **Creating a Service Key** section in [Accessing API Management APIs Programmatically \[page 127\]](#).
- You have fetched a valid bearer token. To know more about obtaining a bearer token, see the **Obtaining a Bearer Token** section in [Accessing API Management APIs Programmatically \[page 127\]](#).

### Procedure

1. Run the services using a standard REST console.
2. Create a new certificate store reference using the below API:
  - Service URL: `https://<url-from-service-key>/apiportal/api/1.0/Management.svc/CertificateStoreReferences`
  - Method: POST
  - Request Header: Authorization: Bearer <token>
  - Content Type: application/json
  - Accept: application/json
  - Request Body:

#### Sample Code

```
{
 "name": "reference-name"
 "certificateStoreName": "store-name"
}
```

#### Note

- name - This is the name of the certificate store reference
- certificateStoreName - This is the name of the certificate store to which this certificate store reference is pointing to

- CertificateStoreReference created successfully  
Response: 201

#### Sample Code

```
{
 "d": {
 "__metadata": {
 "id": "https://<API portal URL>/apiportal/api/1.0/Management.svc/CertificateStoreReferences('reference_ts_1')",
 "uri": "https://<API portal URL>/apiportal/api/1.0/Management.svc/CertificateStoreReferences('reference_ts_1')",
 "type": "apiportal.CertificateStoreReference"
 },
 "certificateStoreName": "ts",
 "life_cycle": {
 "__metadata": {
 "type": "apiportal.History"
 },
 "changed_at": "\/Date(1709793613017)\/",
 "changed_by": "sb-apiaccess1689070958781!b6077|api-portal-xsuaa!b2160",
 "created_at": "\/Date(1709793612858)\/",
 "created_by": "sb-apiaccess1689070958781!b6077|api-portal-xsuaa!b2160"
 },
 "name": "reference_ts_1",
 "storeType": "TRUSTSTORE"
 }
}
```

- When a CertificateStoreReference with the same name already exists  
Response: 400 Bad request:

#### Sample Code

```
{
 "error": {
 "code": "CERTIFICATE_STORE_REFERENCE_NAME_DUPLICATION_ERROR",
 "message": {
 "lang": "en",
 "value": "A certificate store reference already exists with the same name. Please provide a different name and try creating again."
 }
 }
}
```

- When no such certificate store exists with the given name  
Response: 400 Bad request

#### Sample Code

```
{
 "error": {
 "code": "CERTIFICATE_STORE_REFERENCE_CREATE_FAILED_LINKED_CERTIFICATE_STORE_VALIDATION_ERROR",
 "message": {
 "lang": "en",

```

```
 "value": "The certificate store reference cannot be created
as the certificate store name provided is invalid. Provide a correct
certificate store name and try creating again."
 }
}
```

## 1.5.3.4.2 Updating the References

You can use the CertificateStoreReferences API to update a certificate store reference via service calls.

### Prerequisites

- You must have a service key for the **APIPortal.Administrator** role. To know more about creating a service key for accessing APIs in the API portal, see the **Creating a Service Key** section in [Accessing API Management APIs Programmatically \[page 127\]](#).
- You have fetched a valid bearer token. To know more about obtaining a bearer token, see the **Obtaining a Bearer Token** section in [Accessing API Management APIs Programmatically \[page 127\]](#).

### Procedure

1. Run the services using a standard REST console.
2. Create a new certificate store reference using the below API:
  - Service URL: `https://<url-from-service-key>/apiportal/api/1.0/Management.svc/CertificateStoreReferences('<reference-name>')`
  - Method: PUT
  - Request Header: Authorization: Bearer <token>
  - Content Type: application/json
  - Accept: application/json
  - Request Body:

#### Sample Code

```
{
 "certificateStoreName": "updated-store-name"
}
```

#### Note

certificateStoreName - This is the name of the new certificate store to which this certificate store reference will point to.

- CertificateStoreReference updated successfully  
Response: 204 No Content
- When no such CertificateStoreReference entity exists with the given name  
Response: 400 Bad request:

#### Sample Code

```
{
 "error": {
 "code": "NO_SUCH_CERTIFICATE_STORE_REFERENCE_EXISTS",
 "message": {
 "lang": "en",
 "value": "No such certificate store reference exists.
Please check the certificate store reference name and try again."
 }
 }
}
```

- When no such certificate store store exists with the given name  
Response: 400 Bad request:

#### Sample Code

```
{
 "error": {
 "code":
"CERTIFICATE_STORE_REFERENCE_UPDATE_FAILED_LINKED_CERTIFICATE_STORE_VALIDATION_ERROR",
 "message": {
 "lang": "en",
 "value": "The certificate store reference cannot be updated
as the certificate store name provided is invalid. Please provide a
correct certificate store name and try creating again."
 }
 }
}
```

### 1.5.3.4.3 Reading the References

Use the CertificateStoreReference API to get the list of certificate store references.

#### Prerequisites

- You must have a service key for the **APIPortal.Administrator** role.  
To know more about creating a service key for accessing APIs in the API portal, see the **Creating a Service Key** section in [Accessing API Management APIs Programmatically \[page 127\]](#).
- You have fetched a valid bearer token. To know more about obtaining a bearer token, see the **Obtaining a Bearer Token** section in [Accessing API Management APIs Programmatically \[page 127\]](#).

## Procedure

1. Run the services using a standard REST console.
2. Get the list of all the certificate store references using the below API:
  - Service URL: `https://<url-from-service-key>/apiportal/api/1.0/Management.svc/CertificateStoreReferences`
  - Method: GET
  - Request Header: Authorization: Bearer <token>
  - Accept: application/json
  - Fetches the list of all the certificate store referencesResponse: 200

### Sample Code

```
{
 "d": {
 "results": [
 {
 "__metadata": {
 "id": "https://<application-url>/apiportal/api/1.0/Management.svc/CertificateStoreReferences('reference_ts_1')",
 "uri": "https://<application-url>/apiportal/api/1.0/Management.svc/CertificateStoreReferences('reference_ts_1')",
 "type": "apiportal.CertificateStoreReference"
 },
 "certificateStoreName": "ts_1",
 "life_cycle": {
 "__metadata": {
 "type": "apiportal.History"
 },
 "changed_at": "\\Date(1709810822439)\\/",
 "changed_by": "sb-apiaccess1689070958781!b6077|api-portal-xsuaa!b2160",
 "created_at": "\\Date(1709793612858)\\/",
 "created_by": "sb-apiaccess1689070958781!b6077|api-portal-xsuaa!b2160"
 },
 "name": "reference_ts_1",
 "storeType": "TRUSTSTORE"
 },

]
 }
}
```

## 1.5.3.4.4 Deleting the References

Use the CertificateStoreReference API to delete the certificate store reference via service calls.

### Prerequisites

- You must have a service key for the **APIPortal.Administrator** role.  
To know more about creating a service key for accessing APIs in the API portal, see the **Creating a Service Key** section in [Accessing API Management APIs Programmatically \[page 127\]](#).
- You have fetched a valid bearer token. To know more about obtaining a bearer token, see the **Obtaining a Bearer Token** section in [Accessing API Management APIs Programmatically \[page 127\]](#).

### Procedure

1. Run the services using a standard REST console.
2. Delete the certificate store reference using the below API:
  - Service URL: `https://<url-from-service-key>/apiportal/api/1.0/Management.svc/CertificateStoreReferences('<reference-name>')`
  - Method: DELETE
  - Request Header: Authorization: Bearer <token>
  - Accept:: application/json
  - CertificateStoreReference deleted successfully.  
Response: 204: No Content
  - When no CertificateStoreReference entity exists with the given name  
Response: 400 Bad request

#### Sample Code

```
{
 "error": {
 "code": "NO_SUCH_CERTIFICATE_STORE_REFERENCE_EXISTS",
 "message": {
 "lang": "en",
 "value": "No such certificate store reference exists.
Please check the certificate store reference name and try again."
 }
 }
}
```

## 1.5.3.5 Create a Policy Template

Create a policy template add it to an API proxy.

### Prerequisites

- You have a thorough understanding of policies and the various flows they can be attached to. For more information, see [Policies \[page 205\]](#).
- You are familiar with the different types of policies supported by . For more information, see [Policy Types \[page 206\]](#).
- You have the payload of the policy you want to create.

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose ► [Configure](#) ► [APIs](#) ►.  
A list of APIs appears in the catalog.
3. In the list, click the API for which you want to create the policy template.
4. On the details screen, choose [Policies](#).
5. On the [Policy Editor](#) screen, choose ► [Policy Template](#) ► [Create](#) ►.
6. In the [Create Policy Template](#) window, proceed as follows:
  - Enter a name for the template in the [Name](#) field.
  - Enter a description for the template in the [Description](#) field.
  - Select the required policies from the [Policies Available](#) list.

#### ⓘ Note

Please ensure that you don't choose Concurrent Rate Limit policy, as this policy is being decommissioned.

#### ⓘ Note

If there are any default fault rules or a post-client flow available within the API proxy, they will also be appended to the policy template.

7. Choose [OK](#).  
To view the policy template that you have just created, navigate from the API portal home page to ► [Configure](#) ► [APIs](#) ► [->Policy Templates](#).

## Related Information

[Apply a Policy Template \[page 568\]](#)

### 1.5.3.5.1 Apply a Policy Template

Apply a policy template to an API.

#### Prerequisites

- You have a thorough understanding of policies and the various flows they can be attached to. For more information, see [Policies \[page 205\]](#).
- You are familiar with the different types of policies supported by . For more information, see [Policy Types \[page 206\]](#).
- You have the payload of the policy you want to create.

#### Context

You are applying a policy template and want to apply it to an API proxy.




##### Note

If you attempt to apply a policy template that includes a Statistic Collector policy with the Statistic name='clientIP' to an API proxy, the application of the policy template will fail because 'clientIP' is a reserved word in Cloud Foundry API Management analytics. To address this issue, we have introduced a default prefix, "**custom\_**", which will be automatically appended to all custom metric names. For example, if the Statistic name is "clientIP", it will be displayed as "**custom\_clientIP**".

##### Remember

If any changes are made to the existing policy, the API proxy must be redeployed.

#### Procedure

1. Log on to the .
  2. Choose the navigation icon on the left and choose  [Configure](#)  [APIs](#) .
- A list of APIs appears in the catalog.



3. In the list, click the API for which you want to apply the policy template.
4. On the details screen, choose *Policies* .
5. On the Policy Editor screen, choose *Edit->Policy Template ->Apply*.
6. Select the policy templates you want to apply from the *Apply Policy Template* window.  
You can click a template to view all the policies available and also select the required policies in that template.

#### Note

When applying the policy template, any default fault rules or a post-client flow available within the policy template will also be appended to the API proxy.

7. If you want to copy only the policies and not the flows, choose *Copy Only*. Otherwise, choose *Apply* to copy both polices and flows.

## Related Information

[Create a Policy Template \[page 567\]](#)

### 1.5.3.5.2 Update a Policy Template

Update a policy template.

## Prerequisites

- You have a thorough understanding on Policies and the various Flows it can be attached to. For more information, see [Policies \[page 205\]](#).
- You are familiar with the different types of policies supported by . For more information, see [Policy Types \[page 206\]](#).

## Context

You are updating a policy template.

## Procedure

1. Log on to the .

2. Choose the navigation icon on the left and choose **Configure > APIs**.  
A list of APIs appears in the catalog.
3. In the list, choose the API for which you want to update the policy template.
4. On the details screen, choose **Policies**.
5. In the policy editor screen, choose **Policy Template > Update**.
6. In the **Update Policy Template** window :
  - Select the policy template that you want to update, from the **Name** dropdown box.
  - Choose the required policies from the list of policies available.

#### Note

If there are any default fault rules or a post-client flow available within the API proxy, they will also be appended to the policy template.

7. Choose **OK**.

## Related Information

[Create a Policy Template \[page 567\]](#)

[Delete a Policy Template \[page 572\]](#)

### 1.5.3.5.3 Import a Policy Template

Import a policy template.

## Prerequisites

- You have a thorough understanding of policies and the various flows they can be attached to. For more information, see [Policies \[page 205\]](#).
- You are familiar with the different types of policies supported by . For more information, see [Policy Types \[page 206\]](#).

## Context

To import a policy template, proceed as follows:

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.  
A list of APIs appears in the catalog.
3. On the **Configure > APIs** screen, choose **POLICY TEMPLATES**.  
A list of available policy templates appears in the catalog.
4. Choose **Import**
5. In the **Import Policy Template** window, attach the policy template you want to import and choose **OK**.

## Related Information

[Create a Policy Template \[page 567\]](#)

### 1.5.3.5.4 Export a Policy Template

Export a policy template.

## Prerequisites


- You have a thorough understanding of policies and the various flows they can be attached to. For more information, see [Policies \[page 205\]](#).
- You are familiar with the different types of policies supported by . For more information, see [Policy Types \[page 206\]](#).

## Context

To export a policy template, proceed as follows:

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.  
A list of APIs appears in the catalog.

3. On the **Configure > APIs** screen, choose *POLICY TEMPLATES*.  
A list of available policy templates appears in the catalog.
4. In the *Actions* column, select *Export* by choosing the  icon for the policy template you want to export.

## Related Information

[Create a Policy Template \[page 567\]](#)

[Import a Policy Template \[page 570\]](#)

### 1.5.3.5.5 Delete a Policy Template

Delete a policy template.


## Prerequisites

- You have a thorough understanding of policies and the various flows they can be attached to. For more information, see [Policies \[page 205\]](#).
- You are familiar with the different types of policies supported by . For more information, see [Policy Types \[page 206\]](#).

## Context

To delete a policy template, proceed as follows:

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.  
A list of APIs appears in the catalog.
3. On the **Configure > APIs** screen, choose *POLICY TEMPLATES*.  
A list of available policy templates appears in the catalog.
4. In the *Actions* column, select *Delete* by choosing the  icon for the policy template you want to delete.
5. Choose *OK*.

## Related Information

[Create a Policy Template \[page 567\]](#)

### 1.5.3.5.6 Policy Template Structure

During an import or export of a policy template, the policy template follows a pre-defined structure.

Policy Template Structure

Folder Name	Path	Contents
Policy Template Container	\PolicyTemplateContainer	Root folder that contains the FileResource and Policy information.
FileResource	\PolicyTemplateContainer\FileResource	Lists all the scripts attached to the policy. Only Java, Python, and XSL Scripts are supported. Follow the below naming convention: <ul style="list-style-type: none"><li>• Java Script: <b>&lt;JavaScript name&gt;.js</b></li><li>• Python script: <b>&lt;PythonScript name&gt;.py</b></li><li>• XSL script: <b>&lt;XSLScript name&gt;.xsl</b></li></ul>
Policy	\PolicyTemplateContainer\Policy	Contains a list of all available policies. Each policy is available as a separate file with the naming convention <b>&lt;Policy name&gt;.xml</b> .
<b>&lt;policytemplatename&gt;.xml</b>	\PolicyTemplateContainer\ <b>&lt;policytemplate&gt;.xml</b>	Contains the header information of all the available policies.

### 1.5.3.6 API Versioning

Versioning allows the creation and management of multiple releases of an API. You can version an API proxy when you want to improve, upgrade, or customize the functional behavior provided by a currently existing API proxy.

#### Prerequisites

You need the APIPortal.Administrator role to use the versioning feature.

A new version of the API proxy is created when:

- Incompatible changes, such as structural changes or changes in payload, are planned.

- Additional policies are enforced, and a step-by-step client transition is needed.
- API developer wants to model changes, which will be exposed as new version without affecting the consumption of existing API version .
- There's change in the billing model.

You can add a version to an API, when you're creating the API proxy. This applies to creation from:

- 
- Import of API definition file to the
- Creation from API designer
- Creation from APIs available from the SAP API Business Hub

Multiple versions of the API, can coexist at both runtime and design time.

The pattern for the name and base path for an API Proxy, when you're using the version attribute, and have chosen, for example, v1 as the version is:

Attribute	Value
Version	v1
Name	Name_v1
Base Path	/v1/SalesOrder

In case you don't provide name and base path in this required pattern, the system appends the version to the name, and prepends the version to the base path, to create a unique version of the API proxy.

For the version, we recommend that you use alphanumeric values. Based on the version you've provided, the system appends the version value to the API proxy name and base path, creating a unique version.

#### → Remember

Once you've versioned an API Proxy, you can't edit the version or the base path.

## Related Information

[Creating a Versioned API \[page 575\]](#)

[Create an API Proxy \[page 478\]](#)

[Create an API Proxy Using the API Designer \[page 488\]](#)

[Copy an API Proxy \[page 581\]](#)

[Import an API Definition \[page 534\]](#)





## 1.5.3.6.1 Creating a Versioned API

Creating a versioned API Proxy from a deployed, versioned, or nonversioned API Proxy in the API Management, API Portal.

### Context

You can create a versioned API Proxy, from either a previously versioned API or a nonversioned API that have been deployed, from the API Portal. To know more about creating a versioned API Proxy when creating it from scratch in the API Portal, see [API Versioning \[page 573\]](#).

### Procedure

1. Log on to the API portal.
2. From the navigation bar, choose  (Develop).  
A list of APIs appears in the catalog.
3. Select the API from which you want to create a new versioned API. This can be either already versioned API, or a nonversioned API.
4. From the top-right corner of the *Overview* screen, choose   *New Version* .
5. In the *Create New Version* screen, in the *Version* field, enter a new version and choose *Create*. You can't edit the name and the base path. However, once you enter a new version, the version is appended to the name and prepended to the base path to create a unique API Proxy version.  
If the version you enter is v1, the name will be Name\_v1, and the basepath will /v1/SalesOrder.
6. You can then choose to *Save* or *Deploy* the new version of this API Proxy.

## 1.5.3.7 API Revisions

With API revisions, you can make incremental changes to an API proxy without causing any disruption to the deployed API. Access the past changes made to the API proxy, and even restore the API to any of its previous states.

In this table, we've summarized the key features of API versions and API revisions :

## API Revision

Revisions typically include small incremental and compatible changes. For example, adding a property or adding a new resource or a policy to an API proxy. You create revisions when there are changes that don't break the existing consumption flows. API revisions are agnostic of the actual URL that is used for consuming the API. Since the deployed revision is being consumed, you don't have to access it separately. The API proxy URL doesn't change across revisions of an API proxy.

Only one revision of an API proxy exists in the runtime. You can view the different revisions in the design time, and compare the contents of different revisions.

## API Version

An API version helps you to track incompatible changes made to an existing API proxy. For example, you're already consuming version 1 of an API proxy, now in version 2 some incompatible changes are done, like changing the data types of the properties, or removing an existing policy or a property. For such incompatible changes, the newer version is made available for consumption and you can adopt it by migrating to the newer version. Since versions are individually accessible for consumption, the version number appears on the URL as shown here: `http://host/api/v2/customers/123`

Multiple versions of an API can coexist in the runtime for consumption as every version has a different API proxy URL with the version information. In design time, a new API proxy gets created for every version.

[Creating API Revisions \[page 576\]](#)

Create API revisions to make nonbreaking API changes in a safe and controlled manner.

### 1.5.3.7.1 Creating API Revisions

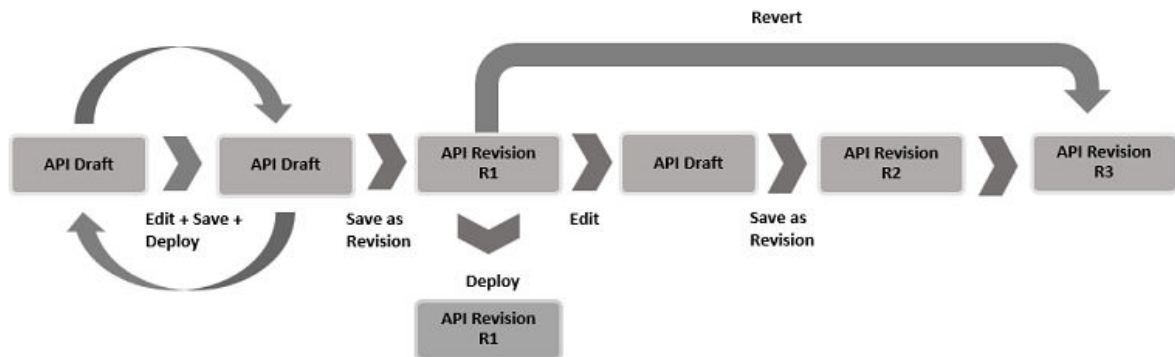
Create API revisions to make nonbreaking API changes in a safe and controlled manner.

#### Context

- When you create and save an API proxy, a draft gets created.
- You can deploy a draft and yet continue to have a working copy of the same.
- You can save this draft as a revision and then deploy this revision.
- Every time you edit and save a revision, a draft gets created.
- You can restore any of the previous revisions using the revert action. The revert action creates a new revision, which is a copy of the previous version you're trying to restore. dialog and choose



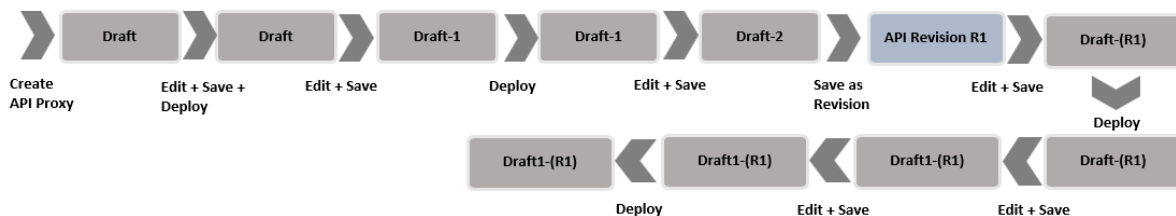
Refer the block diagram to understand the flow:



A unique draft name is generated with each **Deploy** and *Edit-Save* action, following a specific naming pattern. When an API proxy is created, a draft is automatically generated with the name "Draft". If the draft is deployed and then edited, the new draft name will change to *Draft-1* from *Draft*. Subsequent deploy and edit actions will increment the number, resulting in names like *Draft-2* dialog and choose and so on. This naming convention helps to clearly differentiate between the deployed draft and the working draft.

If the draft is originated from a revision, the draft name will include the revision name. For example, if the revision name is "1.0.0", editing and saving this revision will create a draft with the name *Draft-(1.0.0)*. Subsequent deploy and edit actions will increment the number, resulting in names like *Draft-1-(1.0.0)*.

Drafts originating from revisions will include the parent name to ensure a unique name for each API proxy draft.



For visual instructions on how to get started with API Revisions, refer the following [Tutorial](#).

To create a new revision, execute the following steps:

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. Choose *Create*.
4. Fill in the details in the *Create API Create*.

To create an API proxy from scratch, see [Create an API Proxy \[page 478\]](#).

- Select the appropriate tabs to edit the API. You can choose from the following:

Tab	Description
<a href="#">Overview</a>	<p>You can edit the following:</p> <ul style="list-style-type: none"> <li>Name of the API</li> <li>API Base Path</li> <li>API State</li> <li>API Description</li> </ul> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p><b>Note</b></p> <p>You aren't allowed to edit the Host Alias.</p> </div>
<a href="#">Proxy Endpoint</a>	You can add the proxy endpoint and the route rules.
<a href="#">Target Endpoint</a>	You can choose URL, API Provider, or API Proxy, as the target endpoint as well as enter target endpoint rules.
<a href="#">Resources</a>	You can add resources, or change already existing ones.

- Once the changes are made, choose [Save](#).  
A draft gets created under the [Revisions](#) tab.
- To continue to have a working copy of the draft, choose [Edit](#).  
Choose [Save](#) after making all the changes.
- You can also choose to deploy this draft by selecting the **...** icon and choosing the [Deploy](#) option from the list.
- If you want to convert this draft into a revision, select the draft, and choose [Save as Revision](#) from the inline action.
- In the [Save as Revision](#) dialog, provide a revision name and a description, and choose [Save](#).

#### Note

The revision name must be unique and can't be named "Draft" as it's a reserved term. Instead, you can use alphanumeric characters (both lowercase and uppercase), as well as the special characters '\_', ':', '-', and '()'. The name shouldn't exceed 50 characters.

A new revision gets created. You can now edit, delete, and deploy the new revision of the API.

#### Note

If you edit this revision and then save the changes you've made, a draft gets created out of this latest revision.

#### Note

By default, the maximum number of revisions you can create is 20. However, this number is configurable. If you proceed with the [Save as Revision](#) action once you've reached the maximum limit,

the oldest revision (which isn't deployed) gets deleted from the list. However, if the oldest revision is already deployed, you'll not be allowed to save the draft as a revision.

### Note

You can't delete a draft if both the working and the deployed copy of the draft are the same. If only one revision exists at any given point in time, you can't delete the same from the inline actions button. In such a scenario, you need to delete the API proxy. You can do so by choosing the *Delete* option from the *Additional Options* on the top-right corner of the page.

11. If you want to work on any of the previous revisions, select the revision and choose *Revert* from the inline action.

The *Revert* action replaces the existing draft (if any), create a new revision that can be deployed. The new revision is a copy of the previous revision that you're trying to restore.

### Note

API revisions can affect the following actions:

Actions	Behavior
Publishing of products	When you create a product, you link it to one or more APIs. Also, the same API can be linked to multiple products. After you've linked an API to a product, all attributes of the API such as API resources, and API documentation become an implicit part of the product. Now, if the product has an API proxy that has multiple revisions, and the deployed revision of the API proxy isn't the latest revision; if you try to publish such a product, the deployed revision of the API proxy gets published. For more information, see <a href="#">Publish API Proxies [page 652]</a> .
Importing an API proxy	When you import an API, a new revision of the API gets created. If your API has an existing draft, the draft gets replaced by the new revision created during the import. For more information, see <a href="#">Import an API Definition [page 534]</a> .
Transporting APIs and its related artifacts	When you transport an API, a new revision of the API is created. If your API has an existing draft, the draft gets replaced by the new revision created during the transport.  If there are multiple revisions of an API, only the latest revision gets transported. For more information, see <a href="#">Transporting an API Proxy from Source to Destination [page 640]</a> .

## Results

You've created a new revision, created a draft out of the latest revision, used the revert action to create the latest revision of the API proxy from a previous revision.

**Task overview:** [API Revisions \[page 575\]](#)

### 1.5.3.8 API Proxy States

As an API Management administrator, you can set states for an API proxy while creating or updating the API proxy.

The following states can be set for an API proxy:

State	Information
<b>Alpha</b>	A version of an API meant for exploratory purposes.
<b>Beta</b>	A version of an API that isn't meant for productive use.
<b>Active</b>	A version of an API that is meant for productive use.
<b>Deprecated</b>	When an API is marked as deprecated, the customer is encouraged to use a successor API.
<b>Decommissioned</b>	A version of an API where the service is no longer available, and can't be used productively.

When you deprecate or decommission an API proxy, you must provide a deprecation or decommissioning date, as well as a successor API. This could be another API within the or an external link. You can only mark an API proxy as **Deprecated** or **Decommissioned** while updating the proxy and not while creating it.

The API state is to be used only for demarcation, and doesn't play a role in the governance or lifecycle of the API.

#### Note

An application developer can view the states of the deprecated and decommissioned API proxies (if published in the ) in the API details screen of the API business hub enterprise.

## 1.5.3.9 Deploy an API Proxy

After an API is created, you must deploy the API to make it ready for product assignments.

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose ► [Configure](#) ► [APIs](#) ◀.
3. To expose a service as an API, choose [Create](#). See [Create an API Proxy \[page 478\]](#) for the steps on how to create an API.

Once you've filled in all the required details of the API, refer to step 19 in [Create an API Proxy \[page 478\]](#) to [Deploy](#) the API.

#### Note

After deploying the API proxy, there can be instances where the policies or the base path added to the API proxy are modified and saved within the API proxy. In such scenarios, the API proxy gets automatically deployed, and the changes start reflecting over the internet.

### Results

Once an API is deployed, it's available on the internet. The API can be called using the virtual host and the base path and can also be packaged as API Products.

## 1.5.3.10 Copy an API Proxy

You can copy an API proxy in the same subscription.

### Prerequisites

You are assigned with [APIPortal. Administrator](#) role.

## Context

To copy an API proxy, proceed as follows:

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. On the *APIs* tab page, choose the **Action** icon against the required API and then select the *Copy* option. Alternatively, you can open the required API and in the details page select the option *Copy*.
4. In the *Copy API* dialog box, the details for each attribute is pre filled. *Name* and *API Base Path* fields should be unique. So, it is required to change the API name and basepath values. The values for the remaining fields can either be retained or changed.
  - a. Optional: Enter a version for your API proxy.

When you choose to version your API proxy, it's name will be appended with the version, and it's basepath will be prepended with the version. For example, If the version you enter is v1, the name will be Name\_v1, and the basepath will be /v1/SalesOrder. For more information, see [API Versioning \[page 573\]](#)

For more details on each field, see [Create an API Proxy \[page 478\]](#).

5. Choose *Copy*.
6. After you have copied the API, you can select one of the following two actions for the API:

Action	Resulting API State	Future Action on API
<i>Save as Draft</i>	<i>Not Deployed</i> API is available only in the , and is not available for product assignments.	<i>Deploy</i> API is deployed and is ready for product assignments.
<i>Save and Deploy</i>	<i>Deployed</i> Only deployed APIs can be selected for product publishing.	<i>Undeploy</i> If any API is undeployed after being published, it is removed from the developer portal. When the API is deployed again, the product is updated. You can bring down an API without having to delete it from the product assignment. You cannot undeploy an API if it is the only one associated with the product.

### Note

An API proxy consists of a virtual host and a base path. The base path can be identical for multiple API proxies, provided API proxies have different virtual hosts. This means, for an API proxy, the combination of the virtual host and base path should be unique.

The example below explains the same, where AP1 is proxy 1, AP2 is proxy 2, VH1 is Virtual Host 1, VH2 is the Virtual Host 2, and BP(A) is the base path.

Example: AP1 = VH1+ BP(A) AP2 = VH2 + BP(A)

## Related Information

[Import an API Definition \[page 534\]](#)

[Export an API Definition \[page 533\]](#)

[Create an API Proxy Using the API Designer \[page 488\]](#)

### 1.5.3.11 Create a Policy

Define a policy to set rules on the API, for example, to enforce security or control API traffic.

#### Prerequisites

- You have a thorough understanding on Policies and the various Flows it can be attached to. For more information, see [Policies \[page 205\]](#).
- You are familiar with the different types of policies supported by . For more information see, [Policy Types \[page 206\]](#).
- You have the payload of the policy you want to create.

#### Procedure

**Context:** You are creating an API proxy and want to add a policy to it.

1. While creating an API proxy, choose [Policies](#) on the details screen.
2. Select a Flow on which you want to apply the policy.

#### Note

Flows are available in the top left section of the [Policy Designer](#). You can select an existing flow or create a conditional Flow. To create a conditional Flow, choose the icon + beside the [ProxyEndpoint](#) or [TargetEndpoint](#) depending on which endpoint you want to assign the policy. Enter a name for the conditional Flow.

3. From the [Policies](#) section right hand side, choose the icon + ([Add](#)) beside the required policy.
4. In the [Create Policy](#) pop-up, enter a name for your policy.
5. From the [Stream](#) drop-down, select the processing pipeline where this policy should be assigned:

- [Incoming Request](#)
  - [Outgoing Response](#)
6. Choose [Add](#).  
The specified policy is created and denoted in the Policy Designer. A sample payload for the selected policy is added in the editor.
  7. In the [Conditional String](#) field, specify the condition on which this policy should be executed.
  8. In the editor below, provide the payload for the selected policy.
  9. Choose [Save](#).  
The policy now appears under the [Created Policies](#) section. The count beside the Flow to which this policy was assigned, is incremented.  
If required, the policy name can be edited in the policy editor. However, If you edit the policy name, then you need to change the policy name at instances where the policy name is referred.

## 1.5.3.12 Create a Script

This topic describes how to create a FileResource (also called Script).

### Prerequisites

- You are familiar with the concept of Scripts. For more information, see [File Resource \[page 469\]](#).
- You have the payload of the Script that you want to create.

### Context

Script is a FileResource or code snippet that is attached to Flows using policies. . supports the creation of JavaScript, Python, and XSL scripts.

**Context:** You are creating an API proxy and want to add a script to it.

### Procedure

1. While creating an API proxy, navigate to the [Policies on API](#) tab page.
2. Select [Invoke Policy Designer](#).
3. Select the icon + ([Add](#)) beside the [Scripts](#) section at the bottom-left of the Policy Designer.
4. Enter a name for the script.
5. From the [Type](#) field, select one of the following options:
  - [JavaScript](#)
  - [Python](#)



- [XSL](#)

6. Choose [Add](#).

The added script appears under the [Scripts](#) section.

7. Select the Script you created and provide the script details in the editor.

8. Choose [Save](#).

You can reference the scripts from a Java Script policy, Python policy or XSL Transform policy.

### 1.5.3.13 Edit an API Proxy

Once you've created an API proxy you can further change the proxy, either on the , or by using the embedded API designer.

#### Context

When you edit an API proxy either on the or using the API designer, ensure that you save and deploy the API proxy once the changes are made. Saving the API proxy is a design time activity, the changes you've made get pushed to the runtime only when you deploy the changes. Therefore, when you choose [Save](#) after making the changes, the changes are saved locally and don't get published on the API business hub enterprise. Choose [Deploy](#) to perform an explicit deployment to bring in the new changes in the runtime during the API proxy execution.

Consider the following examples:

- If you just save and not deploy the change you've made to the Target Endpoint of an API proxy, and then try to debug the API proxy and use the trace capability in runtime to trace the API call, the call points to the old Target Endpoint. Only when you save and then explicitly deploy the changes, the API call points to the new Target Endpoint.

#### Note

Saving the changes puts the API proxy in the local intermediate save state, only deploying it publishes the changes in runtime.

- Similarly, if you attach new Policies or add new Resources to an API proxy, ensure that you save the changes and then explicitly deploy the proxy for the latest changes to reflect during API Proxy execution in the runtime.
- If an API proxy (that is already part of a published Product and is being consumed via an Application) is changed and those changes are saved and not deployed, then the application runtime doesn't reflect the saved changes.

#### Note

Make sure that the API is deployed before attaching it to a Product. If you try to publish a Product that has an API with saved changes attached to it, the following error message appears: "The API proxy attach to the Product has some changes that aren't deployed yet."

Similarly, if you publish a Product that has multiple APIs attached to it, and few of the APIs have changes that are saved but not deployed, a warning message appears with the lists of APIs that weren't published as the changes weren't deployed.

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. Choose the API you want to edit.  
The *View API* page is displayed. To edit the various tabs available on this screen, choose *Edit* from the top-right corner of the screen.
4. Select the appropriate tabs, to edit the API. You can choose from the following, *Overview*, *Proxy Endpoint*, *Target Endpoint*, and *Resources*.

Tab	Details
Overview	You can edit the following: <ul style="list-style-type: none"> <li>• Name of the API</li> <li>• Host Alias</li> <li>• API Base Path</li> <li>• API State</li> <li>• API Description</li> </ul>
Proxy Endpoint	You can add the proxy endpoint and the route rules.
Target Endpoint	You can choose URL, API Provider, or API proxy, as the target endpoint as well as enter target endpoint rules.
Resources	You can add resources, or change already existing ones.
Revision	Once an API is created, you can plan subsequent compatible changes to the API by creating a revision.

5. To edit the API in the embedded API designer, you can either choose **Edit > Edit in API Designer** from the top-right corner of the screen, or choose **Edit > Resources** tab, click > to open the API designer.
6. You can make required changes to the swagger structure in the API designer. For more information on the API designer, see [API Designer \[page 490\]](#).
7. Once you've made the swagger changes, click *Save*. These changes will then be reflected in the various tabs on the .

### Note

When you're editing the swagger structure in the API designer, editing the same from the tabs is disabled.

If the API proxy is already in the deployed state, then saving the changes after editing the API doesn't deploy the latest changes. Similarly, if the API proxy is already published on the API business hub

enterprise, the save action doesn't publish the latest changes. In both cases, a message appears on the [View API](#) page that the changes you've made aren't deployed. For the changes to reflect during API proxy runtime flow, choose [Click to Deploy](#) and provide your confirmation on the popup window.

## Results

The changes you've made to the API are saved and deployed successfully.

### 1.5.3.14 Delete an API Proxy

To delete an API proxy, you must disassociate the API from the product or delete the product it is associated to.

## Context

In this section, we have described how to delete the API proxy after disassociating it from the product. Alternatively, you can delete the product and then delete the API proxy. However, to delete the product, you must disassociate the product from all the applications where it is being used.

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose [Configure > APIs](#).
3. Choose the API you want to delete.  
The [View API](#) page is displayed.
4. Scroll down to the [Products Associated](#) section, and choose the product.  
You are redirected to the [View Product](#) page.
5. Navigate to the [APIs](#) tab and choose [Edit](#).
6. To disassociate the API from the product, select the checkbox next to the API and choose [Remove](#).  
Provide your confirmation by selecting [Yes](#) on the [Remove API](#) dialog.
7. Navigate back to the [Configure > APIs](#) page and search for the API you removed from the product.
8. To delete the API, choose the [Action](#) icon and choose [Delete](#).
9. Provide your confirmation on the [Delete API](#) dialog.

## Results


Your API proxy is deleted.

### 1.5.3.15 Externally Managed APIs

You can manage APIs that are not managed by but by some other API Management solution.

You can import an API definition (Open API Specification 2.0 or 3.0) of externally managed APIs in , without adding management capability (proxy creation and policies) of and publish them to API business hub enterprise.

#### Features of Externally Managed APIs

- Sometimes these APIs are managed by external gateways.
  - API Proxies are not created for these APIs.
  - These APIs are only listed in . No aspect of their life cycle is managed by .
  - Status (Deployed/Not Deployed) is not displayed for these APIs.
- 
- These APIs are represented by the symbol: 
  - If you create a product using only externally managed APIs , your consumers can't view the rate plan for these APIs, nor can they subscribe to the product or use custom attributes for the product in the API business hub enterprise. For more information, see [Create a Product \[page 653\]](#).
  - You can choose to manage the life cycle of these externally managed APIs after listing them. To do so, from the [Overview](#) page of the API, choose [Manage](#).
  - You can list externally managed APIs by importing them. For more information, see [Import an API Definition \[page 534\]](#).

#### 1.5.3.15.1 Adding Externally Managed APIs

To add an externally managed API, you must import the API definition of an externally managed API in .

#### Prerequisites

- You are assigned with [APIPortal. Administrator](#) role.
- To import an externally managed API, you need to have or create an OpenAPI file for it. The OpenAPI file should specify the URL of the API. If you don't have an OpenAPI specification for your API, you can create one using various open-source editors or the API Designer included within . For more

information, see [Create an API Using the API Designer](#). Also, refer to the "Prerequisites" section in [Import an API Definition \[page 534\]](#).

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. On the *APIs* tab page, choose **Create > Import API**.
4. In the *Import API* window:
5. Browse and select the required API definition from your local file system.  
You can attach files of type .zip and .json.
6. To list an externally managed API, choose *List as Externally Managed API*.

This API will only be listed in , and its lifecycle will not be managed.

In order to access externally managed APIs in the API business hub enterprise, you need to add these APIs to a product and then publish the product. For more information on how to create a product, see [Create a Product](#).

### Note

While you can add externally managed APIs to a product, you will not be able to include rate plans or add custom attributes for them.

If your product consists solely of externally managed APIs, users will not be able to subscribe to the product.

## 1.5.3.15.2 Converting Externally Managed APIs to Internally Managed APIs

You can convert an external API, whose lifecycle is managed by an external API Management solution, to an internal API, which is managed by .


## Context


After the conversion, you can create API proxies for these internally managed APIs, import, and publish them and apply policies to them. You can apply all the capabilities to these managed APIs.

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. Identify the externally managed API that you want to convert from the list of APIs.

### Note

The externally managed API is marked with an  icon, and the *Status* column doesn't display the status of the API.

4. Choose the  *Action* icon against the required API and then select the *Manage* option. Alternatively, you can open the required API, and then select the option *Manage*.
5. You can edit the prefilled data of the externally managed API before choosing *Deploy*.

### Note

You can attach policies to the API only after it's deployed.

6. To attach policies, navigate back to the list of APIs on the **Configure > APIs** page, choose the externally managed API that you've converted into an internally managed API.

You can see that the status column now shows the status of the API as Deployed.

7. Open the required API, and choose *Policies* on the details screen. See [Create a Policy \[page 583\]](#) for more information.

## Results

You've converted an externally managed API into an internally managed API and have applied policies to that managed API.

## Next Steps

To import and publish the internally managed APIs, refer the following topics: [Import an API Definition \[page 534\]](#) and [Publish API Proxies \[page 652\]](#)

### 1.5.3.16 Handling URL Redirects in an API Proxy Using Policies

**Symptom:** The API proxy URL redirects to a different target endpoint.

This topic explains how to handle URL redirects in an API proxy using policies.

Let's say you've deployed a simple API proxy, wherein the target endpoint is pointing to the backend service located at the following URL:

<https://services.odata.org/V2/Northwind/Northwind.svc>

For more information on creating an API proxy, see [Different Methods of Creating an API Proxy \[page 477\]](#).

When you click on the API proxy URL, if your browser displays the service URL instead of the proxied URL, it indicates that the service URL that you are trying to access has been moved temporarily or permanently to a new location. When this scenario occurs, doesn't display the proxied URL. It instead displays the redirected URL of the backend service you provided.

You can handle this URL redirects by adding policies to your API proxy. These policies determine how URL direction is handled within .

In this illustration, we provide two approaches for handling URL redirects. One, wherein we use a Fault Raise policy to stop redirection when the backend service you're trying to access is moved temporarily to a new location, and additionally send a response to the client indicating what is the new redirected URL of the service. Two, wherein we use a combination of various policies such as, Extract Variables policy, Service Callout policy, and Assign Message policy to gracefully handle the URL redirection without user intervention.

## Stop URL Redirection Using Fault Raise Policy

Add a Fault Raise policy to your API proxy with the following configuration. Add this policy in the target endpoint (PostFlow/Outgoing Response).

### Sample Code

```
<RaiseFault async="true" continueOnError="false" enabled="true" xmlns="http://www.sap.com/apimgmt">
 <!-- Defines the response message returned to the requesting client -->
 <FaultResponse>
 <Set>
 <!-- Sets or overwrites HTTP headers in the response message -->
 <Headers/>
 <Payload contentType="text/plain"> Sorry for the inconvenience!
The target URL that you have provided has been redirected
to "{response.header.Location}". The URL redirection has been stopped due
to security reasons. Please provide a target URL that doesn't redirect.</
Payload> <StatusCode>500</StatusCode>
 <!-- sets the reason phrase of the response -->
 <ReasonPhrase>Server Error</ReasonPhrase>
 </Set>
 </FaultResponse>
 <IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
</RaiseFault>
```

Add a condition to indicate when this policy must execute. In our illustration, we want to execute this policy only when the backend service URL that you provided is being redirected to a new URL. We achieve this by including the following condition string:

### Sample Code

```
response.status.code=307
```

The above condition string indicates that when the http response code received from a backend service is 307, the policy gets executed. The http response code 307 indicates that the service/resource you want access can't be accessed from its canonical location, but can be accessed from a new temporary location.

If the above policy executes, the client would get the following response, where `{response.header.location}` would actually be replaced with the redirected URL of the backend service you provided.

### Sample Code

```
Sorry for the inconvenience!
 The target URL that you have provided has been redirected to
 "{response.header.Location}". The URL redirection has been stopped due to
 security reasons. Please provide a target URL that doesn't redirect.
```

Similarly, if the backend service that you want to access is permanently moved to a new location, then you change the condition string statement to `response.status.code=302`.

## Handle URL Redirection Gracefully

Add an Extract Variables policy to your API proxy with the following configuration. Add this policy to the proxy endpoint (PreFlow/Outgoing Response).

### Sample Code

```
<ExtractVariables async="true" continueOnError="false" enabled="true"
 xmlns='http://www.sap.com/apimgmt' >
 <Source>response</Source>
 <Header name="Location">
 <Pattern ignoreCase="true">https://{redirectUrl}/{redirectUrlPath}</
 Pattern>
 </Header>
</ExtractVariables>
```

Add a condition to indicate when this policy must execute. In our illustration, we want to execute this policy only when the backend service URL that you provided is being redirected to a new URL. We achieve this by including the following condition string:

### Sample Code

```
response.status.code=307
```

If the above policy executes, the value of the Location header tagged to the response is extracted and stored in the `redirectUrl` and `redirectUrlPath` variables. For example, if the backend service URL (redirected URL) you're trying to access is `https://services.odata.org/V2/Northwind/Northwind.svc/`, then the value `services.odata.org` is stored in the `redirectUrl` variable and the value `V2/Northwind/Northwind.svc/` is stored in the `redirectUrlPath` variable.

Use the extracted `redirectUrl` and `redirectUrlPath` variables in the Service Callout policy as described further.



Add a Service Callout policy to your API proxy with the following configuration. Add this policy to the proxy endpoint (PreFlow/Outgoing Response).

### Sample Code

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ServiceCallout async="false" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <Request clearPayload="true" variable="myRequest">
 <IgnoreUnresolvedVariables>false</IgnoreUnresolvedVariables>
 </Request>
 <Response>calloutResponse</Response>
 <HTTPTargetConnection>
 <URL>https://{redirectUrl}/{redirectUrlPath}</URL>
 </HTTPTargetConnection>
</ServiceCallout>
```

Add a condition to indicate when this policy must execute. In our illustration, we want to execute this policy only when the backend service URL that you provided is being redirected to a new URL. We achieve this by including the following condition string:

### Sample Code

```
response.status.code=307
```

If the above policy executes, the response message received from the external service indicated in the **URL** element of the policy, is stored in the **calloutResponse** variable. For example, if the values stored in **redirectUrl** and **redirectUrlPath** variables are **services.odata.org** and **V2/Northwind/Northwind.svc/**, respectively, then the response from the URL (redirected URL) **https://services.odata.org/V2/Northwind/Northwind.svc/** will be stored in the **calloutResponse** variable.

In the next step, send the response message obtained from the service callout policy to the client using the Assign Message policy as shown further.

Add an Assign Message policy to your API proxy with the following configuration. Add this policy to the proxy endpoint (PreFlow/Outgoing Response).

### Sample Code

```
<AssignMessage async="false" continueOnError="false" enabled="true"
xmlns='http://www.sap.com/apimgmt'>
 <Set>
 <Payload>{calloutResponse.content}</Payload>
 </Set>
 <IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
 <AssignTo createNew="true" type="response">response</AssignTo>
</AssignMessage>
```

If the above policy executes, the client receives the response message stored in the **calloutResponse** variable.

Save the API proxy after adding the above policies. You can also create a policy template with the above policies so that you can easily attach the policy template to your API proxies to gracefully handle URL redirection. For more information on policy templates, see [Create a Policy Template \[page 567\]](#).

## 1.5.3.17 Enable Streaming of Requests and Responses in an API Proxy

Configure an API proxy to enable HTTP request and response streaming.

### Context

By default, disables streaming of request and response payloads. The payloads are stored directly into a buffer before API proxy pipeline picks them for processing. With streaming disabled, the policies that operate on the payloads work as expected. You can alter this behavior by enabling streaming. When you have enabled streaming, the API proxy pipeline processes the request and response payloads as-is and streams them without any modifications to the client application and to the target endpoint.

You can enable streaming if your API proxy handles large request and response payloads. In , message payload size is restricted to 10 MB for non-streamed HTTP requests and responses. For streamed requests and responses, the message payload size is restricted to 500 MB.

#### ⓘ Note

When you have enabled streaming, recommends that you don't attach policies that require access to the request and response payloads. These policies can cause errors or initiate buffering, which limits the payload size and hence defeats the purpose of enabling streaming for handling large payloads. You can attach policies such as Authentication or message logging policies since these policies don't interact with the request and response payloads.

### Procedure

1. To enable request streaming, in your API proxy bundle, add the `request.streaming.enabled` property in the proxy endpoint and target endpoint definitions and set it to `true`.
2. To enable response streaming, in your API proxy bundle, add the `response.streaming.enabled` property in the proxy endpoint and target endpoint definitions and set it to `true`.

You can locate the proxy endpoint and target endpoint definition files in your API proxy bundle under `APIProxy/APIProxyEndpoint` and `APIProxy/APITargetEndPoint`, respectively.

The following sample code shows how to enable both request and response streaming in the target endpoint definition:

#### ↗ Sample Code

```
<TargetEndPoint>
 <name>default</name>
 <url>https://www.concursolutions.com/api/v3.0/</url>
 <provider_id>NONE</provider_id>
 <isDefault>true</isDefault>
 <properties>
 <property>
 <name>request.streaming.enabled</name>
```

```

 <value>true</value>
 </property>
 <property>
 <name>response.streaming.enabled</name>
 <value>true</value>
 </property>
 </properties>
</TargetEndPoint>

```

The following sample code shows how to enable both request and response streaming in the proxy endpoint definition:

### Sample Code

```

<ProxyEndPoint default="true">
 <name>default</name>
 <base_path>/concur/api/v3.0</base_path>
 <properties>
 <property>
 <name>request.streaming.enabled</name>
 <value>true</value>
 </property>
 <property>
 <name>response.streaming.enabled</name>
 <value>true</value>
 </property>
 </properties>
</ProxyEndPoint>

```

You can also enable streaming directly from the API portal as follows:

#### Enabling Streaming via API portal

1. In the , navigate to the **Configure > APIs** tab.
2. Select the API for which you want to enable streaming.
3. Choose *Edit* from the top-right corner of the screen.
4. Choose *Proxy EndPoint*.
5. Under *Proxy Endpoint Properties*, choose *Add*.
6. Enter the name of the properties you want to add. In this case, add the `request.streaming.enabled` and `response.streaming.enabled` properties and set their value to `true`.
7. Choose *Target EndPoint*.
8. Under *Target Endpoint Properties*, choose *Add*.
9. Enter the name of the properties you want to add. In this case, add the `request.streaming.enabled` and `response.streaming.enabled` properties and set their value to `true`.
10. Save the changes.

## 1.5.3.18 Enable Dynamic Routing

Define route rules to enable dynamic routing in an API proxy.

### Context

A route connects an API proxy endpoint to an API target endpoint. It governs the path of a request from proxy endpoint to target endpoint and determines which target endpoint to invoke based on the condition defined in proxy EndPoint definition. Typically, a route includes a URL used to access the API proxy endpoint and a URL of the backend service defined in target endpoint definition.

In , when you create an API proxy, a default route rule is set and it always forwards the request to the default target endpoint defined in target endpoint definition. When more than one target endpoint is defined, the route rule evaluates the condition set in proxy endpoint definition. If the condition evaluates to true, it forwards the request to the named target endpoint.

The following procedure describes how to achieve dynamic routing in . Let's say you want to route an API proxy request to two different target endpoints, a default target endpoint and a new target endpoint based on a condition set in the proxy endpoint definition.

For our implementation, let's consider the following two target endpoints:

- Target\_Endpoint\_1 (default)  
`https://services.odata.org/V2/Northwind/Northwind.svc/`
- Target\_Endpoint\_2  
`https://services.odata.org/V2/OData/OData.svc/`

### Procedure

#### Creating a simple API proxy

1. Navigate to .
2. Choose the navigation icon on the left and choose ► **Configure** ► **APIs** ►.
3. Under **APIs**, choose **Create** to create a simple API proxy.
4. In the **Create API** wizard, choose the **URL** radio button.

#### Note

You can also choose to create an API by choosing the **API Provider** option. For more information, see [Create an API Proxy \[page 478\]](#)

5. In the **URL** field, enter the target URL of your backend service. In this case, URL pointing to Target\_Endpoint\_1 (default).
6. Enter a name and a title for your API proxy. In this case, let's enter the API proxy name as `Dynamic_Routing`.
7. Scroll down the wizard and enter the base path of your API proxy in the **API Base Path** field. In this case, let's enter the base path as `/multitargets`.

8. Choose [Create](#).
9. Save and deploy your API proxy.

#### Note

When the API proxy is created, the default route rule is set. It points to the default target endpoint and no rule is attached to it.

### Steps for defining new target endpoint

10. Navigate to the **Configure > APIs** tab. From the *APIs* list, choose the API proxy that you deployed.
11. Download the newly deployed API proxy using the [Export](#) option. For more information, see [Export an API Definition \[page 533\]](#)

A zip file called `Dynamic_Routing.zip` is downloaded.

12. Unzip the `Dynamic_Routing.zip` file.

A parent folder called `APIProxy` is created. The `APIProxy` folder consists of various subfolders and files. For more information, see [API Proxy Structure \[page 469\]](#).

13. Open the `APITargetEndPoint` subfolder.

You see a file named `default.xml`. The `default.xml` file contains the URL of the default target endpoint.

14. Create a new XML file named `Target_EndPoint_2.xml` with the following content. In the `Target_EndPoint_2.xml` file, you need to enter a name and the URL of the new target endpoint to which the request must be routed dynamically.

#### Note

The `<isDefault>` attribute must be set to `false` for all the new target endpoints that you define. Whereas, for the default target endpoint the `<isDefault>` attribute would by default be set to `true`.

#### Sample Code

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<TargetEndPoint>
 <name>Target_EndPoint_2</name>
 <url>https://services.odata.org/V2/OData/OData.svc</url>
 <provider_id>NONE</provider_id>
 <isDefault>false</isDefault>
 <properties/>
 <faultRules/>
 <preFlow>
 <name>PreFlow</name>
 </preFlow>
 <postFlow>
 <name>PostFlow</name>
 </postFlow>
 <conditionalFlows/>
</TargetEndPoint>
```

You see two files named `default.xml` and `Target_EndPoint_2.xml` in the `APITargetEndPoint` subfolder.

### Steps for defining conditions using Route Rule

15. Open the `APIProxyEndPoint` subfolder.

You see a file named `default.xml` file.

16. Open the `default.xml` file.

The `default.xml` file contains information about your API proxy such as base path, flows, policies and, the default route rule.

#### Sample Code

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ProxyEndPoint default="true">
 <name>default</name>
 <base_path>/multitargets</base_path>
 <properties/>
 <routeRules>
 <routeRule>
 <name>default</name>
 <targetEndPointName>default</targetEndPointName>
 <sequence>1</sequence>
 <faultRules/>
 </routeRule>
 </routeRules>
 <faultRules/>
 <preFlow>
 <name>PreFlow</name>
 </preFlow>
 <postFlow>
 <name>PostFlow</name>
 </postFlow>
 <conditionalFlows/>
</ProxyEndPoint>
```

17. Update the value of the `<sequence>` attribute to 2.

#### Sample Code

```
<routeRule>
 <name>default</name>
 <targetEndPointName>default</targetEndPointName>
 <sequence>2</sequence>
 <faultRules/>
</routeRule>
```

The resulting `default.xml` file must reflect the following content.

#### Sample Code

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ProxyEndPoint default="true">
 <name>default</name>
 <base_path>/multitargets</base_path>
 <properties/>
 <routeRules>
 <routeRule>
 <name>default</name>
 <targetEndPointName>default</targetEndPointName>
 <sequence>2</sequence>
 <faultRules/>
 </routeRule>
 </routeRules>
 <faultRules/>
 <preFlow>
 <name>PreFlow</name>
```

```

 </preFlow>
 <postFlow>
 <name>PostFlow</name>
 </postFlow>
 <conditionalFlows/>
 </ProxyEndPoint>

```

18. Define a new route rule named `Target_EndPoint_2` by adding the following content to the default.xml file.

#### Sample Code

```

<routeRule>
 <name>Target_EndPoint_2</name>
 <targetEndPointName>Target_EndPoint_2</targetEndPointName>
 <sequence>1</sequence>
 <faultRules/>
</routeRule>

```

The resulting default.xml file must reflect the following content.

#### Sample Code

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ProxyEndPoint default="true">
 <name>default</name>
 <base_path>/multitargets</base_path>
 <properties/>
 <routeRules>
 <routeRule>
 <name>Target_EndPoint_2</name>
 <targetEndPointName>Target_EndPoint_2</targetEndPointName>
 <sequence>1</sequence>
 <faultRules/>
 </routeRule>
 <routeRule>
 <name>default</name>
 <targetEndPointName>default</targetEndPointName>
 <sequence>2</sequence>
 <faultRules/>
 </routeRule>
 </routeRules>
 <faultRules/>
 <preFlow>
 <name>PreFlow</name>
 </preFlow>
 <postFlow>
 <name>PostFlow</name>
 </postFlow>
 <conditionalFlows/>
</ProxyEndPoint>

```

19. Define a condition based on which you want to route the request dynamically. In this case, let's add a `proxy.pathsuffix MatchesPath` condition under the `Target_EndPoint_2` Route Rule and set it to the path called `/Categories`.

#### Sample Code

```

<routeRule>
 <name>Target_EndPoint_2</name>
 <conditions>proxy.pathsuffix MatchesPath "/Categories"</
conditions>

```

```

 <targetEndPointName>Target_EndPoint_2</targetEndPointName>
 <sequence>1</sequence>
 </faultRules/>

```

The resulting default.xml file must reflect the following content.

### Sample Code

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ProxyEndPoint default="true">
 <name>default</name>
 <base_path>/multitargets</base_path>
 <properties/>
 <routeRules>
 <routeRule>
 <name>Target_EndPoint_2</name>
 <conditions>proxy.pathsuffix MatchesPath "/Categories"</
conditions>
 <targetEndPointName>Target_EndPoint_2</targetEndPointName>
 <sequence>1</sequence>
 <faultRules/>
 </routeRule>
 <routeRule>
 <name>default</name>
 <targetEndPointName>default</targetEndPointName>
 <sequence>2</sequence>
 <faultRules/>
 </routeRule>
 </routeRules>
 <faultRules/>
 <preFlow>
 <name>PreFlow</name>
 </preFlow>
 <postFlow>
 <name>PostFlow</name>
 </postFlow>
 <conditionalFlows/>
</ProxyEndPoint>

```

### Note

If you have defined more than one route rule in the proxy endpoint as shown in the above codeblock, their sequence in the XML configuration is important. The first Route Rule to match gets executed. (Route rules with no condition always match). In the above codeblock, if the default route rule appeared first, it would be executed even if the condition of the Target\_EndPoint\_2 route rule would have matched. Hence, it is always recommended to list your conditional route rules before an unconditional route rule.

20. Open the <API\_Proxy\_Name>.xml file. In this case, Dynamic\_Routing.xml file.

21. Add the new target endpoint name that you defined.

### Sample Code

```

<targetEndpoints>
 <targetEndPoint>Target_EndPoint_2</targetEndPoint>
 <targetEndPoint>default</targetEndPoint>
</targetEndpoints>

```

## Steps for viewing dynamic routing



22. Compress the `APIProxy` parent folder.
23. Navigate to API portal and import the compressed `APIProxy.zip` file. For more information, see [Import an API Definition \[page 534\]](#).
24. Choose the imported API proxy.
25. Under *Proxy EndPoint* tab, in the *Route Rules* section, you must see two Route Rules that you defined earlier.

#### Note

You can also add route conditions directly in API portal User Interface instead of adding it manually in the API proxy EndPoint definition file as shown in step 19.

26. Click on the API proxy URL.

The request must be routed to the default target endpoint.

27. Append `/Categories` to the API proxy URL in your browser.

The request must be routed dynamically to the new target endpoint.

To validate the response, copy and paste the actual URL of the backend service with path suffix `/Categories`.

The response obtained must match the response obtained in step 27.

#### Note

All the policies that you attach in the target endpoint via the API portal user interface are applied only to the default target endpoint. In case, if you need to enforce policies on the non-default target endpoint, then you must import the API proxy bundle and manually add the policies in the required target endpoint definition file.

## Related Information

<https://blogs.sap.com/2019/06/03/building-a-loopback-api-using-sap-cloud-platform-api-management/>

### 1.5.3.19 Overriding the Default Update Operation for API Proxy of Type OData

When discovering an API via OData API provider, the update operation is generated automatically based on the backend OData version.

For OData V2 backend service, generates PUT operation as the default update operation. For OData V4 backend service, generates PATCH operation as the default update operation. However, you can override this by adding the key through the API designer for swagger 2.0 or Open API 3.0:

#### Sample Code

YAML Format :

```
:
x-sap-default-update-operation: put
x-sap-default-update-operation: patch
JSON Format:
"x-sap-default-update-operation": "put"
"x-sap-default-update-operation": "patch"
```

In case both the update operations are required, you can pass both the operations as an array as shown below:

### Sample Code

```
JSON Format:
"x-sap-default-update-operation": ["put", "patch"]
YAML Format
x-sap-default-update-operation:
 - put
 - patch
```

### Note

Use array if only both the operations are required. You can use the API designer to add this key in the Open API specification under "info".

```
1 openapi: 3.0.0
2 info: ←
3 title: CATALOGSERVICE
4 version: '1'
5 description: >-
6 <p>This service is located at
7 http://localhost/api/</p>
8 x-sap-default-update-operation:
9 - put
10 - patch
11 servers:
12 - url: 'https://155.56.129.175:443/CATALOGSERVICE_Takt_Demo'
13 tags:
```

Once you have made the changes in the Open API specification, ensure that you *Synchronize* the API proxy for the changes to reflect on the Resource operations.

## Related Information

[Create an API Proxy \[page 478\]](#)

## 1.5.3.20 Creating an API Proxy using SAP Cloud Integration API Provider

Create an API proxy from list of APIs that is deployed in a SAP Cloud Integration tenant.

### Prerequisites

- You should have already created an API provider of type SAP Cloud Integration in **Configure > APIs** tab in the . To do so, see [Create an API Provider \[page 538\]](#).

### Context

Instead of consuming the API services directly, application developers can access APIs exposed via API Management. You do so, by creating an API proxy that covers the service you want to expose. The API maps a publicly available HTTP endpoint backend service. Creating this API proxy lets , from the API Management handle the security and authorizations required to protect, analyze, and monitor your services. Here, you can see how to create an API proxy using an Integration Flow or an API from the list of artifacts deployed in SAP Cloud Integration tenant.

Instead of directly consuming API services, application developers can access APIs through API Management. This is done by creating an API proxy that acts as a gateway for the service you want to expose. The API proxy maps a publicly available HTTP endpoint to a backend service. By creating this API proxy, API Management can handle the security and authorizations needed to protect, analyze, and monitor your services. In this guide, you will learn how to create an API proxy using either an Integration Flow or an API from the list of artifacts deployed in your Cloud Integration tenant.

### Procedure

- Log on to the .
- Choose **Configure > APIs** from the left navigation.
- To expose a service as an API, choose **Create**.
- In order to choose an API of the type OData, REST, or SOAP and create an API Proxy, proceed as follows:
  - In the *Create API* dialog, select the *API Provider* radio button.
  - From the API provider dropdown, select an API provider that belongs to the type *Cloud Integration*.

The dropdown list contains the providers that you're connected to. If the provider you need isn't listed, add an API provider of the type *Cloud Integration* from the **Configure > APIs** tab. For more information, see [Create an API Provider](#).

- Choose *Discover*.

A list of Integration Flows and APIs belonging to type OData, REST, or SOAP appears from Cloud Integration.

- d. Choose an API to connect with the API proxy.
- e. Choose *Next*.
- f. In the *Authentication* dialog, select one of the following authentication methods:
  - *Basic* - Basic authentication is a method of user verification. If the API you have selected supports Basic authentication, you need to enter the credentials, which can be either a Username and Password or a Client ID and Client Secret. For more information, see [Setting Up Basic Inbound Authentication with ClientId and Clientsecret from Service Key in the Cloud Foundry Environment](#).

#### Note

Changing the user credentials for the API might affect proxy execution. However, if necessary, you can modify the user credentials after the proxy creation by following the instructions in the note of step (6):

- *Client Certificate* - Client Certificate authentication is another method of user verification. If the API you have chosen supports Client Certificate authentication, you have the option to upload a certificate in the provided section. The certificate should include both the public and private keys, as well as the certificate chain. The uploaded certificate will be stored in a Store, which is a collection of certificates. For more information, see [Setting Up Inbound Client Certificate Authentication, Cloud Foundry Environment](#).

If you choose:

- *Existing Store*:  
The *Existing Store* option refers to a certificate that has already been created and is currently stored in the system. When you select this option, you will have the ability to choose from a list of certificates that have been uploaded in either .p12 or .pfx format. To proceed, simply select the desired certificate from the *Store Name* dropdown menu. However, please note that you will need to provide the password for the store in order to access and utilize the chosen certificate.

#### Note

If you face any issues while importing the .pkcs/.p12 certificates, please consider checking for restricted characters in your password.

If your password for certificates contains any restricted characters such as (! and #), import of such certificates might fail. Therefore, while creating the certificates please choose a password without these restricted characters and try importing again.

1. From the *Store Name* dropdown, choose an existing store.
  2. Type in the respective password for the existing store.
- *New Store*:
    1. Upload a certificate in either .p12 or .pfx format by selecting the *Browse* option. For more information on the certificates, see [here](#).
    2. Enter a unique name for the store in the *Store Name* field.
    3. Enter a unique name for the certificate in the *Name* field.
    4. Set a password for your certificate in the *Password* field.
    5. Choose *Done*.
  - To implement *OAuth2ClientCredentials* authentication for your API deployed in Cloud Integration, you will need to provide the following information:
    - *Client ID*

- [Client Secret](#)
- [Token URL](#)

The Client ID, Client Secret, and Token URL, are those obtained when you configure OAuth authentication, in particular the Client Credentials Grant variant, for Inbound calls from sender systems to the integration platform. For more information, see [Setting Up OAuth Inbound Authentication with Client Credentials Grant](#).

In the [Create API](#) dialog, the [API Details](#) consisting of [Name](#), [Title](#), [Description](#), [Host Alias](#), [API Base Path](#), and [Service Type](#) are auto populated.

g. Choose [Create](#).

An API provider is auto-created with the name that is populated in the [Name](#) field of the [Create API](#) dialog. This auto-created API provider helps in storing the user credentials provided in the [Authentication](#) dialog and connects to the API proxy.

5. A [Create API](#) screen opens with the API proxy name on the top. You can edit the API proxy details, if necessary and choose [Save](#).

#### Note

While creating API proxies for SOAP and REST, API resources aren't autogenerated; you must add them manually.

While creating API proxies for OData API, autogeneration of resources may be possible in some cases.

6. In the top-right corner of the screen, you have the [Deploy](#) and [Delete](#) option. Choosing [Deploy](#) will deploy the API proxy. On the other hand, if you choose [Delete](#), the API proxy will get deleted. It's important to note that selecting [Save](#) will only create the API proxy without deploying it.

#### Note

To modify the credentials after creating the API proxy, please follow these steps::

1. Go to the [Configure > APIs](#) tab and select the API. You'll find the auto-created API provider with the same name as the selected API.
2. Select the [Target EndPoint](#) tab.
3. Choose the link under [API Provider](#). This is where the user credentials are stored.  
You'll get redirected to the [View API Provider > Overview](#) tab.
4. Select the [Connection](#) tab.
5. On the top-right corner of the screen, choose [Edit](#).
6. Modify the user credentials as needed.
7. Finally, click on the [Save](#) button to save the changes.

#### Related Links:

- [Concepts of Secure Communication](#)

## 1.5.3.21 Custom Attributes

This topic describes custom attributes and services. It is also used to create, delete, and update custom attributes for application and product entities.

Custom attributes can be leveraged to influence the runtime behavior of the API proxy execution. It can be set at a product level or at an application level (when application is created by admin on behalf of developer). Custom attributes provide the flexibility to extend the functionality based on attribute value which can be set or read during the API proxy execution flow. These attributes can be accessed during an API call via the following policies: Verify API key, Access token, and Access entity.

For example, if you add attributes for your products, applications and use Verify API key or Access token verification policy in your flow variables, this can enforce any kind of runtime limitations and control functions.

### Personas

Personas

Role	Component	Details
AuthGroup.API.Admin	API Business Hub Enterprise	User assigned with this role can read, create, delete, and update custom attributes for application.
APIPortal.Service.Catalog.Integration	API Portal	User assigned with this role can read, create, delete, and update custom attributes for application.
APIPortal.Administrator	API Portal	User assigned with this role can read, create, delete, and update custom attributes for products

### Limits

Limits

Attribute	Value
Custom attribute name size	255 characters
Custom attribute value size	1024 characters
Number of custom attributes permitted	18

### Sample Payload

Sample payload to create a custom attribute

- Url: `https://<consumer API-Portal host>:<port>/apiportal/api/1.0/Management.svc/APIProducts HTTP/1.1`
- Method: POST

- Content type: application/JSON
- If you are in the Neo environment, fetch the x-csrf -token:
  - Service url: https://<consumer API-Portal host>:<port>/apiportal/api/1.0/Management.svc/APIProducts HTTP/1.1
  - Method: HEAD
  - Request Header: x-csrf-token: fetch
  - Response: x-csrf-token value.

If you are in the Cloud Foundry environment, fetch the bearer token:

- Service url: https://<consumer API-Portal host>/apiportal/api/1.0/Management.svc/APIProducts HTTP/1.1
- Method: HEAD
- Request Header: Authorization:Bearer <Token for API access>
- Response: bearer-token value

To know how to retrieve this token, see [Accessing API Management APIs Programmatically \[page 127\]](#).

### Sample Code

```
{
 "name": "SampleProduct",
 "version": "1",
 "isPublished": false,
 "status_code": "PUBLISHED",
 "title": "SampleProduct",
 "description": "SampleProduct",
 "isRestricted": false,
 "scope": "",
 "quotaCount": null,
 "quotaInterval": null,
 "quotaTimeUnit": null,
 "additionalProperties": [
 {
 "entityId": "SampleProduct",
 "name": "key1",
 "value": "val1"
 },
 {
 "entityId": "SampleProduct",
 "name": "key2",
 "value": "val2"
 }
],
 "apiProxies": [
 {
 "__metadata": {
 "uri": "APIProxies(name='SampleAPI') "
 }
 }
],
 "apiResources": [],
 "__metadata": {
 "type": "apiportal.APIProduct"
 }
}
```

Sample payload to create a custom attribute (batch call)

### Sample Code

```

Content-Type: application/http
Content-Transfer-Encoding: binary
PUT APIProducts(name='SampleProduct') HTTP/1.1
Request Header: x-csrf-token: <value>
Accept-Language: en-US
Accept: application/json
MaxDataServiceVersion: 2.0
DataServiceVersion: 2.0
Content-Type: application/json
Content-Length: 234
{"name":"SampleProduct","title":"SampleProduct","scope":"","description":"<p>S
ampleProduct</
p>","version":"1","status_code":"PUBLISHED","isRestricted":false,"isPublished"
:true,"quotaCount":-99,"quotaInterval":-99,"quotaTimeUnit":null}
--changeset
Content-Type: application/http
Content-Transfer-Encoding: binary
POST APIProductAdditionalProperties HTTP/1.1
x-csrf-token: fetch
Accept-Language: en-US
Accept: application/json
MaxDataServiceVersion: 2.0
DataServiceVersion: 2.0
Content-Type: application/json
Content-Length: 60
{"entityId": "SampleProduct", "name": "key3", "value": "val3"}

```

Sample payload to update a custom attribute (batch call)

### Sample Code

```

Content-Type: multipart/mixed; boundary=changeset_9c02-68be-2f72
--changeset
Content-Type: application/http
Content-Transfer-Encoding: binary
PUT APIProducts(name='SampleProduct') HTTP/1.1
Request Header: x-csrf-token: <value>
Accept-Language: en-US
Accept: application/json
MaxDataServiceVersion: 2.0
DataServiceVersion: 2.0
Content-Type: application/json
Content-Length: 234
{"name":"SampleProduct","title":"SampleProduct","scope":"","description":"<p>S
ampleProduct</
p>","version":"1","status_code":"PUBLISHED","isRestricted":false,"isPublished"
:true,"quotaCount":-99,"quotaInterval":-99,"quotaTimeUnit":null}
--changeset
Content-Type: application/http
Content-Transfer-Encoding: binary
PUT APIProductAdditionalProperties(entityId='SampleProduct',name='key3')
HTTP/1.1
x-csrf-token: <value>
Accept-Language: en-US
Accept: application/json
MaxDataServiceVersion: 2.0
DataServiceVersion: 2.0
Content-Type: application/json
Content-Length: 14
{"value":"vx"}

```

Sample payload to delete a custom attribute (batch call)

- Url: [https://<consumer API-Portal host>:<port>/apiportal/api/1.0/Management.svc/\\$batch HTTP/1.1](https://<consumer API-Portal host>:<port>/apiportal/api/1.0/Management.svc/$batch HTTP/1.1)



- Method: POST
- Content type: application/JSON
- Request Header: x-csrf-token: fetch (for Neo environment)  
Request Header: Authorization:Bearer <Token for API access> (for Cloud Foundry environment)  
To know how to retrieve this token, see [Accessing API Management APIs Programmatically \[page 127\]](#).

## Sample Code

```
Content-Type: multipart/mixed; boundary=changeset_9c02-68be-2f72
--changeset
Content-Type: application/http
1
Content-Transfer-Encoding: binary
PUT APIProducts(name='SampleProduct') HTTP/1.1
x-csrf-token: Fetch
Accept-Language: en-US
Accept: application/json
MaxDataServiceVersion: 2.0
DataServiceVersion: 2.0
Content-Type: application/json
Content-Length: 234
{"name":"SampleProduct","title":"SampleProduct","scope":"","description":"<p>S
ampleProduct</
p>","version":"1","status_code":"PUBLISHED","isRestricted":false,"isPublished"
:true,"quotaCount":-99,"quotaInterval":-99,"quotaTimeUnit":null}
--changeset
Content-Type: application/http
Content-Transfer-Encoding: binary
PUT APIProductAdditionalProperties(entityId='SampleProduct',name='key3')
HTTP/1.1
x-csrf-token: fetch
Accept-Language: en-US
Accept: application/json
MaxDataServiceVersion: 2.0
DataServiceVersion: 2.0
Content-Type: application/json
Content-Length: 14
{"value":"vx"}
```

Sample payload to create a custom attribute (application)

- Url: https://<consumer Dev-Portal host>:<port>/odata/1.0/data.svc/APIMgmt.Applications HTTP/1.1
- Method: POST
- Content type: application/JSON
- If you are in the Neo environment, fetch the x-csrf -token:
  - Service url: https://<consumer Dev-Portal host>:<port>/odata/1.0/data.svc/APIMgmt.Applications HTTP/1.1
  - Method: HEAD
  - Request Header: x-csrf-token: fetch
  - Response: x-csrf-token value
- If you are in the Cloud Foundry environment, fetch the bearer token:
  - Service url: https://<consumer Dev-Portal host>/odata/1.0/data.svc/APIMgmt.Applications HTTP/1.1
  - Method: HEAD
  - Request Header: Authorization:Bearer <Token for API access>
  - Response: bearer-token value

To know how to retrieve this token, see [Accessing API business hub enterprise APIs Programmatically \[page 134\]](#).

### Sample Code

```
{
 "id": "00000000000000000000000000000000",
 "version": "1",
 "title": "<AppName>",
 "developer_id": "b",
 "ToSubscriptions": [
 {
 "ToAPIProduct": [
 {
 "__metadata": {
 "uri": "APIMgmt.APIProducts(' <ProdName> ')"
 }
 }
],
 "id": "00000000000000000000000000000000"
 }
],
 "ToAttributes": [
 {
 "name": "<AttributeName>",
 "value": "<Attributevalue>",
 "entityType": "Applications",
 "entityId": "0000000"
 }, {
 "name": "<AttributeName>",
 "value": "<Attributevalue>",
 "entityType": "Applications",
 "entityId": "0000000"
 }, {
 "name": "<AttributeName>",
 "value": "<Attributevalue>",
 "entityType": "Applications",
 "entityId": "0000000"
 }
]
}
```

Sample payload to create a custom attribute via navigation (application)

- Url: `https://<consumer Dev-Portal host>:<port>/odata/1.0/data.svc/APIMgmt.Applications(<application_id>)/ToAttributes`
- Method: POST
- Content type: application/JSON
- If you are in the Neo environment, fetch the x-csrf -token:
  - Service url: `https://<consumer Dev-Portal host>:<port>/odata/1.0/data.svc/APIMgmt.Applications(<application_id>)/ToAttributes`
  - Method: HEAD
  - Request Header: x-csrf-token: fetch
  - Response: x-csrf-token value

If you are in the Cloud Foundry environment, fetch the bearer token:

- Service url: `https://<consumer Dev-Portal host>/odata/1.0/data.svc/APIMgmt.Applications(<application_id>)/ToAttributes`
- Method: HEAD

- Request Header: Authorization:Bearer <Token for API access>
- Response: bearer-token value

To know how to retrieve this token, see [Accessing API business hub enterprise APIs Programmatically \[page 134\]](#).

#### Sample Code

```
{
 "name": "<AttributeName>",
 "value": "<Attributevalue>",
 "entityType": "Applications",
 "entityId": "0000000"
}
```

Sample payload to update a custom attribute (application)

- Url: https://<consumer Dev-Portal host>:<port>/odata/1.0/data.svc/APIMgmt.Attributes(name=<attribute\_name>,entityId=<application\_id>,entityType='Applications')
- Method: PUT
- Content type: application/JSON
- Request Header: x-csrf-token: fetch

#### Sample Code

```
{
 "name": "<AttributeName>",
 "value": "<Attributevalue_updated>",
 "entityType": "Applications",
 "entityId": "0000000"
}
```

Sample URL to delete a custom attribute (application)

- Url: https://<consumer Dev-Portal host>:<port>/odata/1.0/data.svc/APIMgmt.Attributes(name=<attribute\_name>,entityId=<application\_id>,entityType='Applications')
- Method: DELETE
- Content type: application/JSON
- Request Header: x-csrf-token: fetch

## Related Information

[Add Custom Attributes to a Product \[page 612\]](#)

## 1.5.3.21.1 Add Custom Attributes to a Product

Add custom attributes to a product.

### Prerequisites

- You are assigned the admin role.

### Context

Use this procedure to add custom attributes to a product.

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and navigate to ► [Configure](#) ► [APIs](#) ▼.
3. Choose [Products](#) and select the product for which you want to add the custom attribute.
4. In the product details page, choose [Custom attributes](#).
5. In the [Custom attributes](#) section, choose [Add](#). Provide a [Name](#) and [Value](#) for the custom attribute. To add more attributes, choose [Add](#).

To delete a custom attribute, choose the delete icon under the [Actions](#) column.

6. Save the changes.

## 1.5.3.21.2 Add Custom Attributes to an Application

You can create applications on behalf of other application developers, add custom attributes to your applications, and manage them.

The custom attributes at application level have values assigned to them. These values help in configuring and accessing them on runtime easily. Here, when admin creates an application on behalf of developer at application level, custom attributes can be assigned by the admin. Therefore, this helps you in enhancing the functionality and performing attribute specific runtime enforcements for your API.

### Role of a API business hub enterprise Administrator:

- Create an application on behalf of a user and handover the application key and secret to that user.
- Create new applications in different landscapes(example: production, non-production) by maintaining the same application key and secret.

- Define custom attributes at application level and regulate the API call logic.

### Visibility of List of Applications

When a user has permission, "ManageAllAPISubscriptions" or is a API business hub enterprise admin, they can view [My Workspace](#) tab on the home page of the API business hub enterprise. After choosing [My Workspace](#) tab, they can view the following changes:

- A list of all the applications created for the tenant by all the developers.
- Navigate to the application details screen by selecting one of the applications. Hence, [My Workspace](#) tab also helps in managing all the applications.

#### Note

You can delete the application that is no longer needed or not in use.

### Creation of Application on Behalf of a User (Application developer)

Creation of application on behalf of a user is only possible if you have the permission "ManageAllAPISubscriptions". Follow the steps mentioned below:

1. Choose [My Workspace](#).
2. In order to create an application, choose the + button next to the [Search](#) field.
3. On the [Create an Application](#) screen, enter the following:
  - [Application Title](#)
  - [Description\(optional\)](#)
  - [Call Back URL](#)
4. Select the checkbox [Create this application on behalf of someone else](#).
5. Mention the [User ID](#) of the user or select it from the filtered drop down list. The drop down list will contain (First\_name, Last\_name and User ID) of the user.
6. If you already have an application key and secret, the checkbox, [Already have Application Key and Secret](#) can be selected. Then the two input boxes with application key and secret appears. Enter the same in the given field.

#### Note

You can reuse the same application key and secret if you have created an application in a different landscape (example: production, non-production).

7. You shall be able to add more products, associated with this application by choosing the + button at the bottom.
8. The checkbox [Take me to this application now](#) is already selected. It can be unchecked, if not necessary.
9. Choose [Save](#) option.

You shall be navigated to your created application, if step 8 is followed. Otherwise an application of your choice can be selected from [My Workspace](#) screen. Therefore:

- In the section [Application Info](#), you can edit name, decription and call back URL.
- Add/Remove products in [Products](#) section.
- Add/Delete/Modify in [Custom Attribute](#) section. You can only modify the value of a custom attribute and not the attribute name.

### Reading Custom Attributes on Behalf of the user

As a user having permission "ReadAllCustomAttributes", you can now see a tab/section where All Custom Attributes for an application will be visible.

Constraints

Attribute	Value
Custom attribute name size for Application	235 characters
Custom attribute value size for Application	1024 characters
Number of custom attributes permitted for Application	18

#### Updating Custom Attributes on behalf of the user

As a user having permission "ManageAllCustomAttributes", you can update the custom attributes for an application and also delete custom attributes .

### 1.5.3.21.3 Restoring Application ID across Landscapes

Maintaining the same application ID across landscapes.

#### Prerequisites

- You have the `AuthGroup.API.Admin` role assigned to you.
- You have the application ID for the application you want to port or recreate.

#### Note

The application ID only contains alphanumeric characters, underscores(\_), and hyphens(-).

#### Context

Each application created in a particular landscape is associated with an application ID. When you are migrating to a new landscape, you can recreate the application there, but this will be associated with a new application ID. To ensure that identical applications are available across landscapes, the original application ID is used via an API provided in this document.

#### Procedure

- Use the POST operation on the following service URL to create an application. The request body to be used for the POST operation is provided in the sample code.

URL:

#### Code Syntax

```
<dev-portal-host>/odata/1.0/data.svc/APIMgmt.Applications
```

- The host changes in the above URL depending on what is used.

#### Sample Code

```
{
 "id": "<application_id>",
 "version": "1",
 "title": "<application_title>",
 "description": "<application_description>",
 "callbackurl": null,
 "developer_id": "<developer_id>",
 "ToSubscriptions": [
 {
 "ToAPIProduct": [
 {
 "__metadata": {
 "uri": "APIMgmt.APIProducts(' <product_name>') "
 }
 }
],
 "id": "00000000000000000000000000000000"
 }
]
}
```

- Mention the application\_title, application\_description, developer\_id and product\_name in the above code along with application id.

Using this API will ensure that applications are recreated or ported to the new landscape with the same application ID and are identical.

## 1.5.3.22 Load Balancing Across API Providers

Load-balancing is configured to distribute the load efficiently across multiple API providers.

Load-Balancing can be applied to an API proxy only when the API proxy is created with a link to an API provider. For more information on how to link an API proxy to the API provider, refer to the [Create an API Proxy \[page 478\]](#).

To perform all the operations related to the target endpoint, for example, fetching the resources and synchronizing all the operations, the API proxy is linked to an API provider. For load balancing, additional API providers are linked to the API proxy.

#### Note

All the API providers that are linked to the API proxy must exist in design time.

#### [Configure Load Balancing During Import \[page 616\]](#)

You can apply load-balancing functionality to an API proxy, by including the load balancer, and health monitor attributes in a .zip file along with the API proxy content.

[Configuring Load Balancing \[page 619\]](#)

You can configure load-balancing functionality for an API proxy from the API Management, API Portal.

## 1.5.3.22.1 Configure Load Balancing During Import

You can apply load-balancing functionality to an API proxy, by including the load balancer, and health monitor attributes in a .zip file along with the API proxy content.

You can attach the .zip file while importing an existing API definition in to the . For more information, refer [Import an API Definition \[page 534\]](#).

You can configure load balancer and health monitor by adding the below attributes to `\APIProxy\APITargetEndPoint\default.xml` in the design time .zip file as shown in the following example:

### Sample Code

```
<additionalAPIProviders>
 <provider_id>target1</provider_id>
 <provider_id>target2</provider_id>
 <provider_id>target3</provider_id>
</additionalAPIProviders>
<loadBalancerConfigurations>
 <maxFailures>5</maxFailures>
 <fallBackServer>target1</fallBackServer>
 <algorithm>RoundRobin</algorithm>
 <serverUnhealthyResponseCode>500,503</serverUnhealthyResponseCode>
 <isRetry>true</isRetry>
 <healthMonitor>
 <intervalInSec>3</intervalInSec>
 <isEnabled>true</isEnabled>
 <httpMonitor>{ "request":
{ "connectTimeoutInSec":18,"socketReadTimeoutInSec":30,"port":443,"verb":"GET",
"path":"/healthcheck"}, "successResponse": {"responseCode":201}}</httpMonitor>
 </healthMonitor>
</loadBalancerConfigurations>
```

The load-balancing attributes in the sample code are defined in the table below:

#### Load Balancer Attributes

Attributes	Definitions
maxFailures	<p>Specifies the number of failed requests from the API proxy to the API provider that results in the request being redirected to another API provider.</p> <p>You must set &lt;MaxFailures&gt; greater than 0 when using the HealthMonitor. When &lt;MaxFailures&gt; value is configured as 0, the Load Balancer tries to connect to the API provider for each request and never removes the API provider from the rotation.</p>



## Attributes

fallBackServer

## Definitions

When all the additional providers fail, then all the requests are sent to this fallback server. When the load balancer determines that all API providers are unavailable, all traffic is routed to the fallback server.

Algorithm

By default *RoundRobin* algorithm is used. But you can also use Weighted and Least Connection algorithms.

The round robin algorithm forwards a request to each API provider in the order in which the API providers are listed in the target endpoint HTTP connection.

The *Weighted* load-balancing algorithm enables you to configure proportional traffic loads for your API providers. The weighted load-balancer distributes request to your API providers in direct proportion to each API provider 's weight. Therefore, the weighted algorithm requires you to set a weight attribute for each API provider as shown in the example below:

### Sample Code

```
<additionalAPIProviders>
 <provider_id>target1</
provider_id>
 <provider_id>target2</
provider_id>
 <provider_id>target3</
provider_id>
</additionalAPIProviders>

<loadBalancerConfigurations>
 <algorithm>Weighted</
algorithm>
 <isRetry>>false</isRetry>
 <fallBackServer>target1</
fallBackServer>

<serverUnhealthyResponseCode>500,50
2,503</serverUnhealthyResponseCode>
<weights>2</weights>
```

In this example, two requests are routed to API providers for every request routed to `<provider_id>target2</provider_id>`.

You can also configure the load-balancer to use the *Least Connection* algorithm. This algorithm routes outbound requests to the API providers with fewest open HTTP connections.

```
<algorithm>LeastConnections</
algorithm>
```

Attributes	Definitions
serverUnhealthyResponseCode	This attribute is added to help ensure that bad HTTP responses, such as 500, increments the failure counter to take an unhealthy server out of load-balancing rotation as soon as possible.
isRetry	If retry is enabled, a request is sent whenever a response failure occurs, for example I/O error, or HTTP timeout. A request is also sent whenever the response received matches a value set by the <serverUnhealthyResponseCode>.

#### HealthMonitor with HTTPMonitor/ TCPMonitor Configuration Attributes

Attributes	Definitions
intervalInSec	The time interval, in seconds, between each polling TCP/ HTTP request.
isEnabled	A boolean that enables or disables the health monitor.
connectTimeoutInSec	Time in which connection to the TCP/HTTP port must be established, to be considered a success. Failure to connect in the specified interval counts as a failure, incrementing the load balancer's failure count for the API provider.
socketReadTimeoutInSec	Time, in seconds, in which data must be read from the HTTP service to be considered a success. Failure to read in the specified interval counts as a failure, incrementing the load balancer's failure count for the API provider.
port	The port on which the HTTP connection to the API provider is established.
verb	Currently, only GET operation is supported. HTTPMonitor submits a GET request to the API provider.
path	The path appended to the URL defined in the API provider. Use this path element to configure a 'polling endpoint' on your HTTP service.
successResponse	Matching options for the inbound HTTP response message generated by the polled API provider. Responses that don't match increment the failure count by 1.
responseCode	The HTTP response code expected to be received from the polled API provider.

**Parent topic:** [Load Balancing Across API Providers \[page 615\]](#)

## Related Information

[Configuring Load Balancing \[page 619\]](#)

## 1.5.3.22.2 Configuring Load Balancing

You can configure load-balancing functionality for an API proxy from the API Management, API Portal.

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose ► *Design* ► *APIs* ◀.

A list of APIs appears in the catalog.

3. Browse for an API proxy, which is already linked to an API provider.
4. Choose the slide button to enable the *Load-Balancing* functionality.
5. Select additional API providers from the *API Provider* dropdown menu.

The *FallBack Server* field appears once the additional API providers are selected.

Select one of the additional API providers as the *FallBack Server* from the dropdown menu.

When the load balancer determines that the additional API providers are unavailable, all traffic is routed to the fallback server.

6. Choose the slide button to enable *Retry*.

If retry is enabled, a request is sent again whenever a response failure occurs, for example I/O error, or HTTP timeout. A request is also sent whenever the response received matches a value set by the Response Code.

7. By default *Round Robin* algorithm is selected. But you can also select *Weighted* and *Least Connection* algorithms.

The **Round Robin** algorithm forwards a request to each API provider in the order in which the API providers are listed in the target endpoint HTTP connection.

The **Weighted** load-balancing algorithm enables you to configure proportional traffic loads for your API providers. The weighted load-balancer distributes request to your API providers in direct proportion to each API provider's weight. Therefore, the weighted algorithm requires you to set a weight attribute for each API provider.

You can also configure the load-balancer to use the *Least Connection* algorithm. This algorithm routes outbound requests to the API providers with fewest open HTTP connections.

8. Choose the slide button to enable *Maximum Failure*.

When enabled, it checks for the number of failed requests from the API proxy to the API provider that results in the request being redirected to another API provider.

*Maximum Failure Value*: Enter the maximum number of failed requests from the API proxy to the API provider. Set the maximum failure value greater than 0 when using the Health Monitor. When the value is configured as 0, the Load Balancer tries to connect to the API provider for each request and never removes the API provider from the rotation.

#### Note

The provider is removed from load balancer configuration, if the maximum number of failed requests is reached.

*Response Code:* Enter the HTTP response codes that are expected to be received from the polled API providers.

9. Choose *Save*.

## Results

You have configured load balancing for the API proxy.

**Task overview:** [Load Balancing Across API Providers \[page 615\]](#)

## Related Information

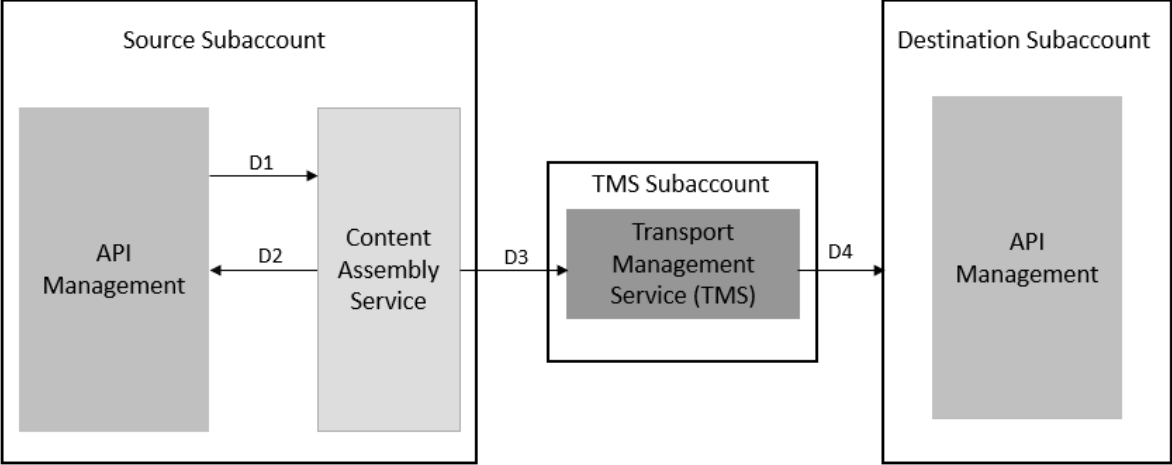
[Configure Load Balancing During Import \[page 616\]](#)

### 1.5.3.23 Transport APIs and Its Related Artifacts

API artifacts and their respective application-specific content, can be reused across multiple tenants using the transport mechanism.

You can use the SAP Cloud Transport Management service (TMS) for exporting, importing, and shipping the APIs and its related artifacts from the development or test environment to production environment. For example, you can design and test API portal content on the test tenant and then use the Cloud Transport Management service to move the content to the target tenant.

This block diagram shows how the content is selected and transported when the Transport Management service is enabled for the first time to transport APIs and its related artifacts from the development or test environment to the production environment:

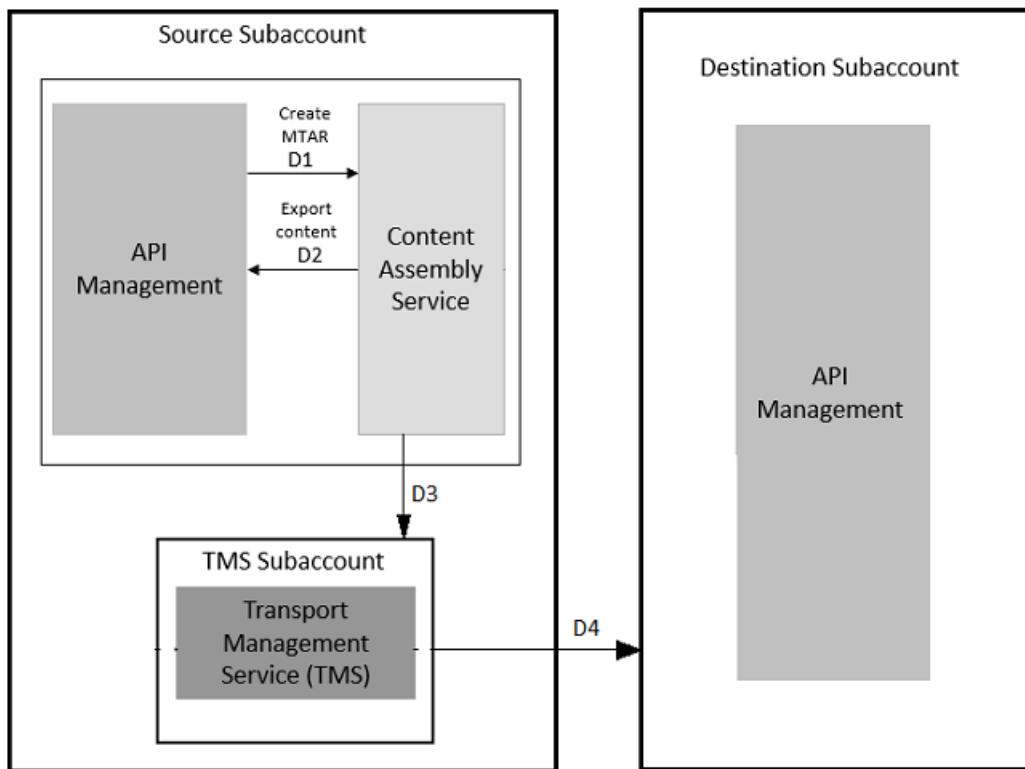


- [Creating Content Assembly Service Destination \[page 627\]](#)
- [Creating API Management Destination \[page 629\]](#)
- [Creating Transport Management Destination \[page 634\]](#)
- [Create a Deploy Service Destination in Cloud Transport Management Service Subaccount \[page 636\]](#)

**Note**

In this case, you have to create a separate subaccount for Transport Management service.

This block diagram shows how the content is selected and transported when the Transport Management service is already enabled for other SAP offerings and you're trying to use it for transporting APIs and its related artifacts from the development or test environment to the production environment:



### Note

In this case, you don't have to create a separate subaccount for Transport Management service. You can continue to use the Transport Management service in the source subaccount.

Once the user initiates transport of the desired API content, the following events take place:

D1- API Management makes an API call to the Content Assembly Service to inform about the transport.

D2 - Content Assembly Service then makes an API call to fetch the API content from workspace. The Content Assembly Service wraps the API content for transport.

D3- The Content Assembly Service makes a second API call to push the API content to the Transport Management service (TMS).

D4- The SAP Transport Management service makes an API call to the Deploy Service. The Deploy Service calls the API Management in the destination subaccount to import the package into the API Management workspace within the Integration Suite.

## 1.5.3.23.1 Enabling Content Transport Using SAP Cloud Transport Management Service

Configure the service instances and destinations, and establish a route between the source and destination nodes to enable transportation of APIs and its related artifacts.

### Prerequisites

- Create a source subaccount and subscribe to API portal, API Management. For more information, see [Set Up API Portal Application \[page 120\]](#).
- Create a **transport** subaccount and subscribe to SAP Cloud Transport Management service. Set up and subscribe to SAP Cloud Transport Management service as described in [Set Up the Environment to Transport Content Archives directly in an Application](#).  
To view and access the SAP Cloud Transport Management service, assign TMS\_ADMIN and TMS\_VIEWER roles to yourself. To set the roles, scroll down to "Steps to Assign User Roles and Permissions" section in [Set Up the Environment to Transport Content Archives directly in an Application](#).
- Create a **Destination** subaccount and subscribe to API portal, API Management. For more information, see [Set Up API Portal Application \[page 120\]](#).

### Context

Let us consider a scenario where you want to transport the API Management content from your source subaccount (which is your Development instance) to your destination subaccount (which is your test instance). You must configure the following in the source and the Transport Management subaccounts as shown in the block diagram:

Source subaccount

- An instance of Content Agent
- An instance of API Management, API Portal
- ContentAssembly Service destination
- API Management, API Portal destination
- Transport Management destination

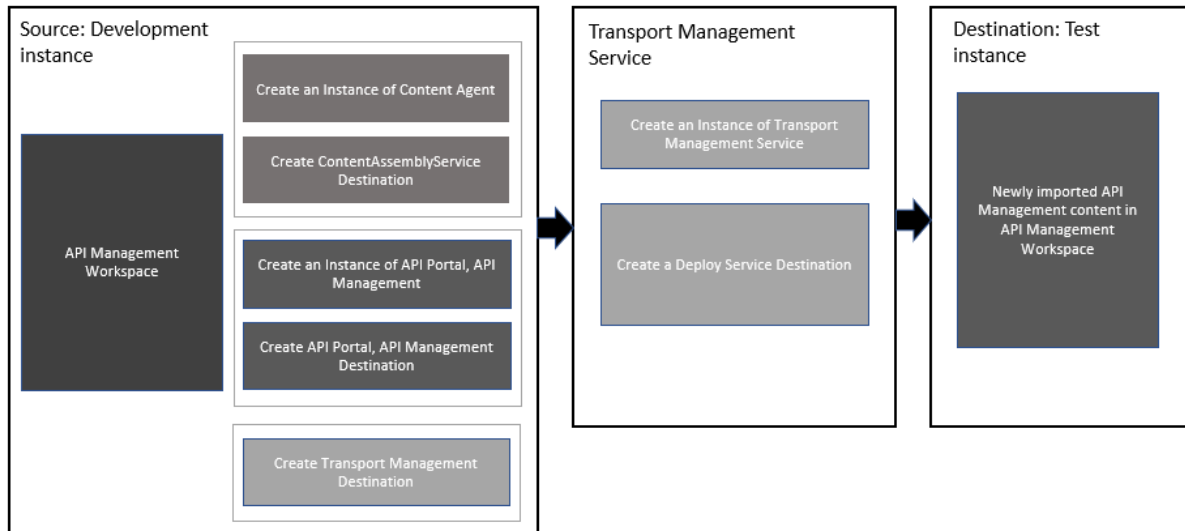
Transport Management service subaccount

- An instance of Transport Management service
- Deploy Service destination

#### 📘 Note

In case the Transport Management service is already enabled for other SAP offerings, and you're trying to use it for transporting APIs and its related artifacts, then you don't have to create a separate subaccount for the Transport Management service. You can continue to use the Transport Management service in your source subaccount.

Further, if you want to transport APIs and its related artifacts from the test to your production instance, your test instance will now become your source. Therefore, you must make all the necessary configurations that you previously made in your development instance in your test instance.



Enabling Content Transport in Cloud Foundry involves the following steps:

Steps	Action	Videos
1	<p>Create an instance of Content Agent.</p> <p>You can transport the API Management artifacts, such as API products, API proxies, KVMs, certificates, and API providers by creating a service instance and a service key of content agent in your source subaccount. See, <a href="#">Create Service Key</a> and <a href="#">Creating an Instance of Content Agent [page 626]</a>. You can also refer to the video in the next column.</p> <div data-bbox="603 1514 986 1908" style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p><b>Note</b></p> <p>Alternatively, you can transport only the API products and the API proxies using the Content Agent service user interface. To do this, you need to <b>subscribe</b> to the free plan of Content Agent application. For more information, see <a href="#">Subscribe to Content Agent Service</a>.</p> </div>	



Steps	Action	Videos
2	<p>Create a ContentAssemblyService destination in source subaccount to make API calls to the Content Assembly Service. See, <a href="#">Create SAP Content Agent Service Destination</a>.</p> <div data-bbox="603 539 991 797" style="border: 1px solid #ccc; padding: 10px; background-color: #f9f9f9;"> <p><b>Note</b></p> <p>If you've already subscribed to the Content Agent service as mentioned in Step 1, you can skip creating destination ContentAssemblyService in source subaccount.</p> </div>	
3	<p>Create an instance of API portal, API Management in your source subaccount and create a service key. See, <a href="#">Creating an Instance of API portal, API Management [page 627]</a></p>	
4	<p>Create destination APIManagement in your source subaccount to make API calls for fetching the API content from the API portal workspace. See, <a href="#">Creating API Management Destination [page 629]</a></p>	
5	<p>Create a service instance and a service key of Transport Management service in your transport subaccount. See, <a href="#">Creating an Instance of SAP Cloud Transport Management Service [page 631]</a></p>	
6	<p>Add a <i>Source</i> node in Transport Management Applications. See, <a href="#">Adding a Source Node in Transport Management Applications [page 633]</a></p>	
7	<p>Create destination TransportManagement in your source subaccount to make API calls to the Transport Management service (TMS). See, <a href="#">Creating Transport Management Destination [page 634]</a></p>	
8	<p>Create a destination in Transport subaccount for the deploy service. See, <a href="#">Create a Deploy Service Destination in Cloud Transport Management Service Subaccount [page 636]</a></p>	
9	<p>Add a Destination node in Transport Management Applications. See, <a href="#">Adding a Destination Node in Cloud Transport Management Applications [page 638]</a></p>	

Steps	Action	Videos
10	Create a transport route to connect the source tenant to the destination tenant. See, <a href="#">Connecting the Source and the Destination Nodes [page 639]</a>	

### 1.5.3.23.1.1 Creating an Instance of Content Agent

Create a service instance and a service key of content agent in your source subaccount. The service key details are needed while creating the **ContentAssemblyService** destination.

#### Context

Content Agent allows you to assemble the content of different content providers, and export it to the transport queue.

#### Procedure

1. Create a service instance of Content Agent. To create the service instance, follow the steps described in [Create Instance](#).
2. Create a service key of Content Agent. For more information, see [Create Service Key](#).

Once the service key is created, make a note of the url, clientid, and clientsecret as these details would be needed while creating HTTP destination ContentAssemblyService for Content Agent. To copy the details, perform the following steps:

1. Choose the Content Agent service instance that you recently created to expand the right-pane.
2. To view the credentials, choose the Content Agent *<Service Key Name>*.
3. Choose the *JSON* tab and copy the URL.

#### Results

You've created an instance of Content Assembly Service and its corresponding service key in the source subaccount.

## 1.5.3.23.1.2 Creating Content Assembly Service Destination

Create go in source subaccount to make API calls to the Content Assembly Service.

### Prerequisites

Create an instance of Content Agent and fetch the service keys. For more information, see [Creating an Instance of Content Agent \[page 626\]](#).

### Procedure

To create the service instance, follow the steps described in [Create SAP Content Agent Service Destination](#).

After creating the destination, you can also do a [Check Connection](#) to verify whether you've added the destination correctly. Once you perform a check connection, the following pop-up message appears:

**Connection to "ContentAssemblyService" is established. Response returned: "401: Unauthorized"**

#### Note

In this case, 401 is an accepted response code.

### Results

You've created the destination [ContentAssemblyService](#).

## 1.5.3.23.1.3 Creating an Instance of API portal, API Management

Create an instance of API portal, API Management in your source subaccount and create a service key. The service key details are needed while creating the **APIManagement** destination.

### Prerequisites

Create an API portal, API Management subaccount and subscribe to it. Set it as your source subaccount, from where you can start exporting the API Management content. For more information, see [Set Up API Portal Application \[page 120\]](#).

## Procedure

1. Create a service instance for API Management, API portal.

Follow the steps described in "Creating a Service Instance in the API Management,API portal" section in [Accessing API Management APIs Programmatically \[page 127\]](#).

### Note

If you don't see the API Management, API portal instance under the *Instances* tab on the *Instances and Subscription* page, then you must add the entitlement.

To add the entitlement:

1. Choose the *Entitlements* tab from the left pane and choose **Configure Entitlements** **Add Service Plans**.
2. Under *Entitlements available for this subaccount*, select API Management,API portal.
3. On the right pane, select the available plans for API Management,API portal and choose *Add ( ) Service Plan*.
4. On the *Entitlements* page, choose *Save*.

2. Create a service key for API portal, API Management.

Follow the steps described in "Creating a Service Key" section in [Accessing API Management APIs Programmatically \[page 127\]](#).

### Note

While configuring JSON, only the *APIportal.Administrator* role must be assigned to you and not the "APIPortal.Guest" and "APIManagement.SelfService.Administrator" roles as mentioned in [Accessing API Management APIs Programmatically \[page 127\]](#).

Once the service key is created, make a note of the url, clientid, clientsecret, and token as these details would be needed while creating HTTP destination *APIManagement*. To copy the details, perform the following steps:

1. Choose the API portal, API Management service instance that you created recently, to expand the right-pane.
2. To view the credentials, choose the *<Service Key Name>*.
3. Choose the *JSON* tab and copy the details.

### Sample Code

```
{
 "url": "https://<apiportal application
name>.cfapps.sap.hana.ondemand.com",
 "tokenUrl": "https://<Space
name>.authentication.sap.hana.ondemand.com/oauth/token",
 "clientId": "sb-apiaccessxxxxxxxx!xxxx|api-portal-xsuaa!bxxxx",
 "clientSecret": "xxxxxxxxxxxxxxxxxxxxxxxxxxxx="
}
```

## Results

You've created an instance and a service key of API portal, API Management in the source subaccount.

### 1.5.3.23.1.4 Creating API Management Destination

Create destination **APIManagement** in your source subaccount to make API calls for fetching the API content from the API portal workspace.

## Prerequisites

Create an instance of API portal, API Management service and fetch the service keys from the service instance as shown in the samplecode. For more information, see [Creating an Instance of API portal, API Management \[page 627\]](#).

### Sample Code

```
{
 "url": "https://<apiportal application
name>.cfapps.sap.hana.ondemand.com",
 "tokenUrl": "https://<Space name>.authentication.sap.hana.ondemand.com/
oauth/token",
 "clientId": "sb-apiaccessxxxxxxxx!xxx|api-portal-xsuaa!bxxxx",
 "clientSecret": "xxxxxxxxxxxxxxxxxxxxxxxxxxxx="
}
```

## Procedure

1. In your web browser, log on to SAP BTP Cockpit and navigate to your source subaccount.
2. Choose the *Destinations* tab in the left-hand pane.
3. Choose *New Destination*.
4. In *Destination Configuration* section, provide values in fields based on description in table.

### Note

Use the Client ID, Client Secret, and the Token Service URL from the Prerequisite section.

Fields	Details
Name	Enter <b>APIManagement</b> as the destination name. Please note that this value is case-sensitive.
Type	Enter <b>HTTP</b> as the supported type.
Description	Enter a brief description stating the purpose of creating a new destination in the <i>Description</i> field.
URL	Provide the URL from the service key details and append <b>/api/1.0/transportmodule/Transport</b> to it.
Proxy Type	Internet
Authentication	Select the authentication type as <b>OAuth2ClientCredentials</b> .
Client ID	Provide the client ID from the service key details.
Client Secret	Enter the client secret.
Token Service URL	Provide the URL from the service key details.

- Choose *Save*.

You can also do a *Check Connection* to verify whether you've added the destination correctly. Once you perform a check connection, the following pop-up message appears: **Connection to "APIManagement" is established.**

#### Note

In case you receive a "401: Unauthorized" response code, please note that its an accepted response code.

## Results

You've created the destination *APIManagement*.

## 1.5.3.23.15 Creating an Instance of SAP Cloud Transport Management Service

Create a service instance and a service key of SAP Cloud Transport Management service in your transport subaccount. The service keys details are needed while creating the **TransportManagement** destination.

### Prerequisites

Create a **Transport** subaccount and subscribe to SAP Cloud Transport Management service. Set up and subscribe to SAP Cloud Transport Management service as described in [Set Up the Environment to Transport Content Archives directly in an Application](#).

To view and access the SAP Cloud Transport Management service, assign TMS\_ADMIN and TMS\_VIEWER roles to yourself. To set the roles, scroll down to "Steps to Assign User Roles and Permissions" section in [Set Up the Environment to Transport Content Archives directly in an Application](#).

### Procedure

1. Create a service instance for SAP Cloud Transport Management service.

To create the service instance, follow step 9 described in [Set Up the Environment to Transport Content Archives directly in an Application](#).

#### Note

If you don't see the *Cloud Transport Management*, instance under the *Instances* tab on the *Instances and Subscription* page, then you must add the entitlement.

To add the entitlement:

1. Choose the *Entitlements* tab from the left pane and choose **Configure Entitlements** **Add Service Plans**.
2. Under *Entitlements available for this subaccount*, search and select *Cloud Transport Management*.
3. On the right pane, select the available plans for Cloud Transport management and choose **Add ( ) Service Plan**.
4. On the *Entitlements* page, choose **Save**.

2. Create a service key for SAP Cloud Transport Management service. To create the service key, follow step 10 described in [Set Up the Environment to Transport Content Archives directly in an Application](#).

Once the service key is created, make a note of the url, clientid, clientsecret, and token as these details would be needed while creating HTTP destination *TransportManagement*. To copy the details, perform the following steps:

1. Choose the service instance *<API Portal, API Management>* that you created recently, to expand the right-pane.
2. To view the credentials, choose the *<Service Key Name>*.

3. Choose the *JSON* tab and copy the details.

#### Sample Code

```
{
 "uaa": {
 "clientId": "sb-ebxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx!bxxxx|alm-ts-backend!bxxx",
 "clientsecret": "xxxxxxxxxxxxxxxxxxxxxxxxxx=",
 "url": "https://<Space name>.authentication.sap.hana.ondemand.com",
 "identityzone": "<Space name>",
 "identityzoneid": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
 "tenantid": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
 "tenantmode": "dedicated",
 "sburl": "https://internal-xsuaa.authentication.sap.hana.ondemand.com",
 "apiurl": "https://api.authentication.sap.hana.ondemand.com",
 "verificationkey": "-----BEGIN PUBLIC KEY-----xxxxxxxx-----END PUBLIC KEY-----",
 "xsappname": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx!bxxxx|alm-ts-backend!bxxx",
 "subaccountid": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
 "uaadomain": "authentication.sap.hana.ondemand.com",
 "zoneid": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
 },
 "uri": "https://transport-service-app-backend.ts.cfapps.sap.hana.ondemand.com"
```

## Results

You've created an instance and a service key of SAP Cloud Transport Management service in the transport subaccount.



## 1.5.3.23.16 Adding a Source Node in Transport Management Applications

The Transport Management Application contains the transported content, so you need a Source node to represent the source endpoint.

### Context

To add a Source node in the Transport Management Applications, execute the following steps:

### Procedure

1. In your web browser, log on to SAP BTP Cockpit and navigate to your Transport subaccount.
2. Choose *Instances and Subscriptions* from the left pane.
3. Go to **► Subscriptions ► Cloud Transport Management ►** and choose *Go To Application*.
4. Choose *Transport Nodes* from the left pane.
5. Choose **+** to add a source node.

#### Note

If the user has a Cloud Integration subscription in the same source subaccount and is opting for SAP Cloud Transport Management service for transporting the content, then the source node can be reused for the Integration suite.

If Cloud Integration and API Management capabilities are activated under Integration suite subscription, then both the capabilities can use the same source node and the destination node.

6. In the *Create Node* dialog box, enter a node name that is unique, for example **Source\_node1**, and enable the *Allow Upload to Node* option.
7. Choose *OK*.

### Results

You've added a Source node in the Cloud Transport Management Applications.

## 1.5.3.23.17 Creating Transport Management Destination

Create destination **TransportManagement** in your source subaccount to make API calls to the SAP Cloud Transport Management service (TMS).

### Prerequisites

Create an instance of SAP Cloud Transport Management service and fetch the service keys from the service instance as shown in the sample code. For more information, see [Creating an Instance of SAP Cloud Transport Management Service \[page 631\]](#).

#### Sample Code

```
{
 "uaa": {
 "clientid": "sb-ebxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx!bxxxx|alm-ts-backend!
 bxxx",
 "clientsecret": "xxxxxxxxxxxxxxxxxxxxxxxxx=",
 "url": "https://<Space name>.authentication.sap.hana.ondemand.com",
 "identityzone": "<Space name>",
 "identityzoneid": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
 "tenantid": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
 "tenantmode": "dedicated",
 "sbaseUrl": "https://internal-xsuaa.authentication.sap.hana.ondemand.com",
 "apiurl": "https://api.authentication.sap.hana.ondemand.com",
 "verificationkey": "-----BEGIN PUBLIC KEY-----xxxxxxxx-----END PUBLIC
 KEY-----",
 "xsappname": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx!bxxxx|alm-ts-backend!
 bxxx",
 "subaccountid": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
 "uaadomain": "authentication.sap.hana.ondemand.com",
 "zoneid": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
 },
 "uri": "https://transport-service-app-backend.ts.cfapps.sap.hana.ondemand.com"
}
```

### Procedure

1. In your web browser, log on to SAP BTP Cockpit and navigate to your source subaccount.
2. Choose the *Destinations* tab in the left-hand pane.
3. Choose *New Destination*.
4. In *Destination Configuration* section, provide values in fields based on description in table.

#### Note

Use the Client ID, Client Secret, and the Token Service URL from the Prerequisite section.

Fields	Details
Name	Enter <b>TransportManagementService</b> as the destination name.
Type	Enter <b>HTTP</b> as the supported type.
Description	Enter a brief description stating the purpose of creating a new destination in the <i>Description</i> field.
URL	Provide the URL from the service key details of the SAP Cloud Transport Management service plan.
Proxy Type	Internet
Authentication	Select the authentication type as <b>OAuth2ClientCredentials</b> .
Client ID	Provide the client ID from the service key details of the SAP Cloud Transport Management service plan.
Client Secret	Enter the client secret.
Token Service URL	Navigate to the SAP Cloud Transport Management Service instance and copy the token service URL from the service key details and append <code>/oauth/token</code> to it.

5. Add an additional property to the SAP Cloud Transport Management service by choosing **Edit** **New Property**.

Add **sourceSystemId** in the first text box.

#### Note

While entering the `sourceSystemId` in the first text box, don't change the case of the text **sourceSystemId**.

Enter the source node that you created in the Transport Management Applications in the second text box. See [Adding a Source Node in Transport Management Applications \[page 633\]](#) for the steps on how to add a Source node.

6. Choose **Save**.

You can also do a **Check Connection** to verify whether you've added the destination correctly. Once you perform a check connection, the following pop-up message appears: **Connection to "TransportManagementService" is established.**

#### Note

In case you receive a "401: Unauthorized" response code, please note that its an accepted response code.

## Results

You've created the destination *TransportManagement*.

### 1.5.3.23.18 Create a Deploy Service Destination in Cloud Transport Management Service Subaccount

Create a destination **TMSDeploy** in Transport subaccount for the deploy service. The deploy service calls the API Management in the destination subaccount to transport the API content into the API Management workspace.

## Prerequisites

- You've already created a Destination subaccount and have subscribed to API portal.
- Create a *Space* in Destination subaccount by:
  1. Choosing *Create Space*.
  2. Choose a name and assign *Space Manager* and *Space Developer* roles to the user.
- Ensure that API portal API Access Plan is enabled for the API Management Instance creation in the Destination subaccount.

## Context

Import request is delegated to deploy-service using destination D4.

## Procedure

1. In your web browser, log on to SAP BTP Cockpit and navigate to Transport subaccount.
2. Choose the *Destinations* tab in the left-hand pane.
3. Choose *New Destination*.
4. In *Destination Configuration* section, provide values in fields based on description in table.:

Fields	Details
Name	Enter a unique name for this destination, for example, <b>TMSDeploy</b> .

Fields	Details
Type	Enter <b>HTTP</b> as the supported type.
Description	Enter a brief description stating the purpose of creating a new destination in the <i>Description</i> field.
URL	<p>Enter <b>https://deploy-service.cfapps.&lt;default-domain&gt;/slprot/&lt;myorg&gt;/&lt;myspace&gt;/slp</b> in the URL field.</p> <p>For <b>&lt;myorg&gt;</b> details navigate to your subaccount in BTP cockpit and choose <i>Overview</i> on the left pane. Choose <i>Cloud Foundry Environment</i> on the <i>Overview</i> page and copy the <i>Org Name</i> that appears in this section.</p> <div style="border: 1px solid #ccc; background-color: #f9f9f9; padding: 10px; margin: 10px 0;"> <p><b>Note</b></p> <p>If your Org Name has spaces, add escape characters in place of spaces, for example, SAP BTP should be written as SAP%20BTP.</p> </div> <p>For <b>&lt;myspace&gt;</b> details navigate to your subaccount in BTP cockpit and choose <i>Overview</i> on the left pane. Choose <i>Cloud Foundry Environment</i> on the <i>Overview</i> page and copy the <i>Space name</i> that appears in this section. This is the space inside the subaccount that has the Cloud Transport Management instance subscribed.</p>
Proxy Type	Internet
Authentication	Select the authentication type as <b>BasicAuthentication</b> .
User ID	Specify the ID of the technical user that is used for the deployment.
Password	Specify the password of the technical user.

5. Choose *Next*.

Choose *Check Connection* to verify whether you've added the destination correctly. Once you perform a check connection, the following pop-up message appears: **Connection to "TMSDeploy" is established.**

## Results

You've created the destination D4 TMS\_Destination\_DeployService. You've also created the target node Destination\_node1.

## 1.5.3.23.19 Adding a Destination Node in Cloud Transport Management Applications

The Transport Management Application contains the transported content, so you need a Destination node to represent the target endpoint.

### Context

To add a Destination node in the Cloud Transport Management Applications, execute the following steps:

### Procedure

1. In your web browser, log on to SAP BTP Cockpit and navigate to your Transport subaccount.
2. Choose *Instances and Subscriptions* from the left pane.
3. Go to ► *Subscriptions* ► *Cloud Transport Management* ► and choose *Go To Application*.
4. Choose *Transport Nodes* from the left pane.
5. Choose + to add a destination node.

#### Note

If the user has a Cloud Integration subscription in the same source subaccount and is opting for SAP Cloud Transport Management service for transporting the content, then the destination node can be reused for the Integration suite.

If Cloud Integration and API Management capabilities are activated under Integration suite subscription, then both the capabilities can use the same source node and the destination node.

6. In the *Create Node* dialog box, enter a node name that is unique, for example **Destination\_node1**, and enable the *Allow Upload to Node* option.
7. Choose **Multi-Target Application** from the *Content Type* dropdown.
8. Choose the destination **TMSDeploy** you created in the Transport subaccount from the *Destination* dropdown. This destination contains the details of the destination subaccount's org and space name. Refer [Create a Deploy Service Destination in Cloud Transport Management Service Subaccount \[page 636\]](#) for the Destination name.
9. Choose *OK*.

### Results

You've added a Destination node in the Transport Management Applications.

## 1.5.3.23.1.10 Connecting the Source and the Destination Nodes

Create a transport route to connect the source tenant to the destination tenant.

### Prerequisites

- You have configured source and destination nodes. For more information, see [Creating Transport Management Destination \[page 634\]](#) and [Create a Deploy Service Destination in Cloud Transport Management Service Subaccount \[page 636\]](#).
- **Administrator** or **LandscapeOperator** roles must be assigned to the user who has subscribed to the TMS application. For more information, see [Security](#)

### Procedure

1. Choose *Transport Routes* from the left pane.
2. Choose **+** to add a transport route.
3. Enter the name of the transport route, and a description.
4. Choose the source and the destination nodes from the existing transport nodes.

#### Note

This destination node is created for the destination in the Transport subaccount and contains the details of the destination subaccount's org and space name.

5. Choose *OK*.

### Results

The transport route is created. You've established a connection between the source and the destination node.

## 1.5.3.23.2 Triggering Content Transport Using SAP Cloud Transport Management Service

After configuring the system for transport, you can start transporting the API proxy and the API artifacts from the source to the destination.

### Context

When transport is triggered, the API proxy and its related artifacts, such as API provider, Key Store Certificate, Trust Store, and Key Value Maps is transported to the destination. However, you can also trigger the transport of each of these artifacts individually.

#### [Transporting an API Proxy from Source to Destination \[page 640\]](#)

When transport is triggered for an API proxy, all artifacts of the API proxy get transported along with the API.

#### [Transporting an API Provider from Source to Destination \[page 642\]](#)

You can choose to transport a single API provider from the source to the destination API portal. When you transport an API provider, it gets created in the destination.

#### [Transporting a Certificate from Source to Destination \[page 643\]](#)

You can choose to transport a single Key Store Certificate or a Trust Store Certificate from the source API portal to the destination API portal.

#### [Transporting a Key Value Map from Source to Destination \[page 644\]](#)

You can transport a single Key Value Map from the source API portal to the destination API portal.

#### [Transporting a Product from Source to Destination \[page 646\]](#)

You can choose to transport a single product from the source to the destination API portal. When you transport a product, it gets created in the destination.

#### [Editing the Security Fields for the Imported API Entities \[page 647\]](#)

To use the imported API entities, the security fields for these entities, which carried dummy values during the import due to security reasons must be replaced with the actual values.

### 1.5.3.23.2.1 Transporting an API Proxy from Source to Destination

When transport is triggered for an API proxy, all artifacts of the API proxy get transported along with the API.

### Context

API proxies always get imported to the destination API portal in the deployed state.



### Note

If the API proxy and its artifacts already exist in the destination, only the API proxy gets overwritten during transport. All other artifacts of the API proxy, such as API provider, Key Store Certificate, Trust Store, and Key-Value Maps remain unaffected.

### Note

When you transport an API, a new revision of the API is created. If your API has an existing draft, the draft gets replaced by the new revision created during the transport.

If there are multiple revisions of an API, only the latest revision gets transported.

### Note

Consider the following use case for transporting API proxies associated with multiple target endpoints:

- **Case 1- Multiple target endpoints in target endpoint XML**  
This customization in the API proxy is achieved by exporting the proxy zip and modifying the target endpoint XML. In such scenarios, when the proxy is transported, the related API providers are automatically created in the target, followed by proxy creation.
- **Case 2- API provider is referenced in some policy of the API proxy**  
The above use case is not supported for transport. In such cases, the provider referred to in the policy must be transported first, followed by the API proxy transport.

In exceptional cases where a proxy with identical Name and Base path is present in both the source and target, but one of them is a versioned API while the other is not versioned, the transfer to the target will fail because conversion between the two is not permitted.

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. Choose the **API** that you want to transport on the **APIs** tab page.
4. Choose the **Action** icon against the required API and then select the **Transport** option. Alternatively, you can open the required API and in the details page select the option **Transport**.
5. On the **Transport** popup, provide a description and choose **Yes**.

The reference number for the transport request gets generated.

In the Transport workbench, you can search for this API in the destination node under the **Transport Description**.

6. Go to the Transport subaccount and perform the following steps to navigate to the **Transport Nodes**.
  - a. In your web browser, log on to SAP BTP Cockpit and navigate to your Transport subaccount.
  - b. Choose **Instances and Subscriptions** from the left pane.
  - c. Go to **Subscriptions > Cloud Transport Management** and choose **Go To Application**.
  - d. Choose **Transport Nodes** from the left pane.

7. Select the destination node, which points to the destination API portal.
8. Under the *Transport Description* column of the destination node, search for the API for which you triggered the transport.
9. Choose *Import Selected* to import the selected API in the queue to the destination node.

## Results

The API content from the sourceAPI portal is transported to the destination API portal.

## Next Steps

Once the API content is transported to the destination API portal, you must ensure that the dummy security values that got imported along with the API entity must be replaced with the actual values. For more information, see [Editing the Security Fields for the Imported API Entities \[page 647\]](#).

## 1.5.3.23.2 Transporting an API Provider from Source to Destination

You can choose to transport a single API provider from the source to the destination API portal. When you transport an API provider, it gets created in the destination.

## Context

### Note

While transporting the API provider, if there are any associated certificates (Key Store or Trust Store), those would get transported to the destination API portal as well.

If you transport an API provider that already exists in the destination API portal, the API provider doesn't get overwritten.

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose ► *Configure* ► *APIs* ◀.
3. Choose the API providers that you want to transport to the *API Providers* tab page.

4. Choose the *Action* icon against the required API provider and then select the *Transport* option. Alternatively, you can open the required API provider and in the details page select the option *Transport*.
5. On the *Transport* popup, provide a description and choose *Yes*.

The reference number for the transport request gets generated.

In the Transport workbench, you can search for the API provider in the destination node under *Transport Description*.

6. Go to the Transport subaccount and perform the following steps to navigate to the *Transport Nodes*.
  - a. In your web browser, log on to SAP BTP Cockpit and navigate to your Transport subaccount.
  - b. Choose *Instances and Subscriptions* from the left pane.
  - c. Go to ► *Subscriptions* ► *Cloud Transport Management* ► and choose *Go To Application*.
  - d. Choose *Transport Nodes* from the left pane.
7. Select the destination node, which points to the destination API portal.
8. Under the *Transport Description* column of the destination node, search for the API provider for which you triggered the transport.
9. Choose *Import Selected* to import the selected API provider in the queue to the destination node.

## Results

Go to the API portal of the destination subaccount, and choose ► *Configure* ► *APIs* ►. Under the *API Providers* tab, look for the API provider you transported. The API provider you transported appears on the list.

### 1.5.3.23.2.3 Transporting a Certificate from Source to Destination

You can choose to transport a single Key Store Certificate or a Trust Store Certificate from the source API portal to the destination API portal.

## Context

### ⓘ Note

Only the certificate with dummy content gets transported from the source to the destination API portal. If you transport a Certificate that already exists in the destination API portal, the Certificate doesn't get overwritten.

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose ► [Configure](#) ► [APIs](#) ▾.
3. Choose the [Certificate](#) that you want to transport on the [Certificates](#) tab page.
4. Choose the [Action](#) icon against the required Certificate and then select the [Transport](#) option. Alternatively, you can open the required Certificate and in the details page select the option [Transport](#).
5. On the [Transport](#) popup, provide a description and choose [Yes](#).

The reference number for the transport request gets generated.

In the Transport workbench, you can search for the Certificate in the destination node under [Transport Description](#).

6. Go to the Transport subaccount and perform the following steps to navigate to the [Transport Nodes](#).
  - a. In your web browser, log on to SAP BTP Cockpit and navigate to your Transport subaccount.
  - b. Choose [Instances and Subscriptions](#) from the left pane.
  - c. Go to ► [Subscriptions](#) ► [Cloud Transport Management](#) ▾ and choose [Go To Application](#).
  - d. Choose [Transport Nodes](#) from the left pane.
7. Select the destination node, which points to the destination API portal.
8. Under the [Transport Description](#) column of the destination node, search for the Certificate for which you triggered the transport.
9. Choose [Import Selected](#) to import the selected Certificate in the queue to the destination node.

## Results

Go to the API portal of the destination subaccount, and choose [Configure](#). Under the [Certificates](#) tab, look for the Certificate you transported. The certificate with dummy content appears on the list.

### 1.5.3.23.2.4 Transporting a Key Value Map from Source to Destination

You can transport a single Key Value Map from the source API portal to the destination API portal.

## Context

For encrypted Key Value Maps, the Key-Value Map is transported with dummy values from the source to the destination API portal. For Non-encrypted Key Value Maps, the Key-Value Map is transported as is.

## Note

If you transport a Key Value Map that already exists in the destination API portal, the Key Value Map doesn't get overwritten.

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. Choose the *Key Value Map* that you want to transport on the *Key Value Maps* tab page.
4. Choose the *Action* icon against the required Key Value Map and then select the *Transport* option. Alternatively, you can open the required Key Value Map and in the details page select the option *Transport*.
5. On the *Transport* popup, provide a description and choose *Yes*.

The reference number for the transport request gets generated.

In the Transport workbench, you can search for the Key Value Map in the destination node under *Transport Description*.

6. Go to the Transport subaccount and perform the following steps to navigate to the *Transport Nodes*.
  - a. In your web browser, log on to SAP BTP Cockpit and navigate to your Transport subaccount.
  - b. Choose *Instances and Subscriptions* from the left pane.
  - c. Go to **Subscriptions > Cloud Transport Management** and choose *Go To Application*.
  - d. Choose *Transport Nodes* from the left pane.
7. Select the destination node, which points to the destination API portal.
8. Under the *Transport Description* column of the destination node, search for the Key Value Map for which you triggered the transport.
9. Choose *Import Selected* to import the selected Key Value Map in the queue to the destination node.

## Results

Go to the API portal of the destination subaccount, and choose *Configure*. Under the *Key Value Maps* tab, look for the Key Value Map you transported. The Key Value Map appears on the list.

## 1.5.3.23.2.5 Transporting a Product from Source to Destination

You can choose to transport a single product from the source to the destination API portal. When you transport a product, it gets created in the destination.

### Context

APIs, Permissions, and custom attributes, that are associated with the product get transported along with the product and get created in the destination API portal. However, the Rate Plan associated with the product doesn't get transported.

#### Note

If the product and its associated entities (APIs, Permissions, and Custom Attributes) already exist in the destination API portal, they get overwritten during the transport. Additionally, the APIs attached to the product get imported to the destination API portal in the deployed state. However, the artifacts of the APIs associated to the product (such as API Provider, Key Store Certificate, Trust Store, and Key-Value Maps) remain unaffected if they already exist in the destination API portal.

Also, transporting a product from the source to the destination API portal in a draft state is not allowed if the product already exists in the destination API portal in a published state and vice versa.

While transporting a product that already exists in the destination portal, ensure that the product is in the same state in both the source and the destination API portal. If the product is in the same state, it gets overwritten in the destination during the transport.

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose ► [Configure](#) ► [APIs](#) ▾.
3. Choose the [Product](#) that you want to transport on the [Products](#) tab page.
4. Choose the [Action](#) icon against the required product and then select the [Transport](#) option. Alternatively, you can open the required product and in the details page select the option [Transport](#).
5. On the [Transport](#) popup, provide a description and choose [Yes](#).

The reference number for the transport request gets generated.

In the Transport workbench, you can search for this product in the destination node under the [Transport Description](#).

6. Go to the Transport subaccount and perform the following steps to navigate to the [Transport Nodes](#).
  - a. In your web browser, log on to SAP BTP Cockpit and navigate to your Transport subaccount.
  - b. Choose [Instances and Subscriptions](#) from the left pane.
  - c. Go to ► [Subscriptions](#) ► [Cloud Transport Management](#) ▾ and choose [Go To Application](#).

- d. Choose *Transport Nodes* from the left pane.
7. Select the destination node, which points to the destination API portal.
8. Under the *Transport Description* column of the destination node, search for the product for which you triggered the transport.
9. Choose *Import Selected* to import the selected product in the queue to the destination node.

## Results

The product from the source API portal is transported to the destination API portal.

## Next Steps

Once the product is transported to the destination API portal, you must ensure that the dummy security values that got imported along with the API entity associated with the product must be replaced with the actual values. For more information, see [Editing the Security Fields for the Imported API Entities \[page 647\]](#).

### 1.5.3.23.2.6 Editing the Security Fields for the Imported API Entities

To use the imported API entities, the security fields for these entities, which carried dummy values during the import due to security reasons must be replaced with the actual values.

## Context

In the destination subaccount, for the imported API entity, replace the dummy certificate attached to the API provider with a genuine certificate. Replace the dummy Key Value Map (KVM) values with authentic values. For different Connection types, replace the dummy values in the username and password fields with valid details.

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. Replace the dummy certificate in the API provider with a genuine certificate.
  1. Choose the API provider and choose *Connection*.
  2. Make a note of the certificate name displayed under *Key Store Certificate* and *Trust Store*.

3. Navigate back to the ► [Configure](#) ► [APIs](#) ▾ page, and look for the certificate on the [Certificates](#) tab.
  4. Delete the certificate, and with the same name create a new certificate. For more information, see [Manage Certificates \[page 558\]](#).
4. Replace the dummy value in the encrypted Key Value Map that got imported along with the API proxy with authentic values:
    1. Navigate to the ► [Configure](#) ► [APIs](#) ▾ page, and choose the [Key Value Maps](#) tab.
    2. Choose the imported Key Value Map and choose [Edit](#).
    3. Delete the dummy value from the [Value](#) field and enter an authentic value.

### Note

For nonencrypted Key Value Maps, dummy values aren't imported. You can use the value of the nonencrypted Key Value Map as is.

5. To replace the dummy values in the [Open Connector](#), [Cloud Integration](#), [On Premise](#), and [Internet](#) type of API providers, perform the following steps:
  - [Open Connector](#) : Choose the API provider , navigate to the [Connection](#) tab, and choose [Edit](#). Replace the dummy values in the [Organization Secret](#) and [User Secret](#) fields with real values.
  - For [Cloud Integration](#) type:
    - If authentication type is selected as [Basic](#), then remove the dummy values and add valid information in the [Username](#) and [Password](#) fields.
    - If authentication type is selected as [OAuth2ClientCredentials](#), then replace the dummy values in the [Client ID](#), [Client Secret](#), and [Token URL](#) fields with real values.
    - If authentication type is selected as [ClientCertificate](#), then make a note of the certificate names displayed under [Key Store Certificate](#) and [Trust Store](#). Navigate back to the [Configure](#) page, look for the certificate on the [Certificates](#) tab. Delete the certificate, and with the same name create a new certificate.
  - In case of [On Premise](#) and [Internet](#) type of API providers, if [Authentication type](#) is set as [Basic](#) in [Catalog Service Settings](#), then you've to remove the dummy values and add valid information in the [Username](#) and [Password](#) fields.

## Results

All the dummy values that got imported along with the API proxy during the transport has been replaced with authentic values. You can start using the imported API proxy.

## 1.6 Test API Proxies

Use the API Test Console to test the runtime behavior of the API proxies.

provides an [API Test Console](#), which enables you to test your API proxies. Testing an API proxy is essential to understand the runtime behavior of the API proxies. The test console allows you to explore the resources associated with an API proxy and execute the operations.



The API Test Console allows you to test OData and REST-based services.

## Procedure

**Context:** You have logged on to the API portal or API business hub enterprise.

1. Log on to the .
2. From the navigation bar on the left choose the **Test > APIs**.
3. A list of APIs appears on the left.

### Note

An API Admin has access to both unpublished and published API proxies, whereas an App Developer can only view a list of published API proxies.

4. Select the required API.  
The URL for the selected API is populated automatically in the *API Test Console*. For the selected API, the URLs of the supported resources appear in the dropdown list. One resource is selected by default.
5. If you want to choose a different collection, use the dropdown list to select the required collection
6. If you have the URL of the service that contains the API proxy, enter the service URL.
7. Choose *Authentication* to select the required type of authentication. You can choose from the following options:
  1. *None*: No authentication required.
  2. *Basic Authentication*: Provide a user name and password.
8. Enable the required method:
  - *GET*: Reads an entity
  - *POST*: Creates an entity
  - *PUT*: Updates an entity
  - *DELETE*: Deletes an entity

### Note

You can enable only the methods supported by the service.

9. Enter the *Request Body* for PUT and POST methods.
10. Choose *Header* to add a header.

### Note

If you want to add multiple headers, choose *Add Request Headers*.

11. Choose *Url Params* to enter the query parameter and value.

### Note

If you want to add multiple query parameters, choose the button *Add URL Params*. Test Console supports passing of custom headers such as X-sap-apimgt-proxy-host:-proxy-trai and X-sap-apimgt-proxy-port:- 8080

12. Choose *Send*.

The response appears in the tabs:

- *Body*: View the formatted response.
- *Body (Raw)*: View the unformatted response.
- *Headers*: View the headers.
- *Cookies*: View the cookies.

13. If you want to use the response body as an input request, choose *Use as Request* on the *Body (Raw)* tab.

14. To view the transactions based on the testing activity that you did, choose *Launch API Viewer*. For more information on tracing API proxy, see [Debug an API Proxy \[page 650\]](#)

## 1.6.1 Debug an API Proxy

You debug an API proxy to troubleshoot and monitor them in , by probing the details of each step through the API proxy flow.

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Configure > APIs**.
3. Select the required API that you want to debug.
4. Choose *Debug* in the View API page.
5. Alternatively, you can also launch the debug viewer from the API Test Console by following the substeps below:
  - a. Select the navigation icon and choose *Test*.
  - b. Select the required API from the APIs list.
  - c. Choose *Debug* at the bottom right corner.
6. In the *API Debug Viewer*, choose *Start Debug*.

When you start the debug, the API records details of each step in the processing pipeline. While the debug session is running, messages and contextual data are captured from live traffic.

#### Note

One debug session supports 20 request/response transactions per message processor through the selected API proxy. A debug session automatically stops after 10 minutes if you don't manually stop it. You can also start the debug session at any point in time of working in the API Management.

You can view a list of captured request/response transactions on the left menu. Choose any of the transactions to view the detailed debug map and the corresponding properties.

7. To view the transaction details at any point in time of an active debug session, choose *Refresh*.
8. When you've captured a sufficient number of requests, choose *Stop Debug*.
9. Debug the API using the guidance provided below:

#### Transaction Map details

Icon	Description
	Indicates a condition evaluated on the API
<b>Condition</b>	
	Indicates the change of state of the execution flow
<b>State Change</b>	
	Indicates the information about the current flow
<b>Flow Information</b>	
	Indicates the result of a condition execution
<b>Execution</b>	
	Indicates an occurrence of error at the time of policy execution
<b>Error</b>	

In addition to the above mentioned icons, each policy is represented by an icon. By choosing the icon, you can view the policy details.

### Phase Details

Using phase details you can check the headers that are being sent to the backend, variables set by policies and so on. You can verify the base path to ensure that a policy is routing the message to correct server. Refer the table below to understand each phase:

Phase	Description
<b>Proxy Endpoint</b>	Indicates the selected proxy Endpoint flow for execution. An API proxy can have multiple named proxy endpoints
<b>Request Headers</b>	Lists the HTTP request headers
<b>Request Content</b>	Displays the HTTP request body

Phase	Description
Variables Read	Lists the flow variables that were read by a policy
Variables Read and Assigned	Lists the flow variables that were read and assigned a value by a policy.
Target Endpoint	Indicates the selected Target Endpoint for execution
Response Headers	Lists the HTTP response headers
Response Content	Displays the HTTP response body

## 1.7 Publish API Proxies

To make your API consumable by external application developers, it is necessary to publish API proxies. Publishing allows you to expose the API proxies in a structured manner, presenting them as a product. To publish API proxies effectively, it is important to understand how to bundle them together and present them as a cohesive product.

A product is a bundle of API proxies. It contains metadata specific to your business for monitoring or analytics. For example, all API proxies related to CRM can be bundled as one CRM product. Instead of publishing API proxies individually, it is easier to bundle related API proxies together as a product and publish it. After including the required API proxies to a product, the product is published to the catalog, where the product is available for Application developers to browse through.

### → Remember

To publish the product, all the API proxies in the product should have a deployed revision.

If the product has one API associated to it, you should ensure that this API is deployed before publishing the product.

Let us assume that a product is associated with five API proxies, out of which three are deployed. When you publish such a product, only three proxies will get published.

You've created multiple revisions out of an API proxy, and deployed one of the revisions. However, the deployed revision is not the latest revision. Now if you attach such an API to a product, and try to publish the product, the deployed revision of the API proxy gets published.

Let us consider another scenario where resources are attached to your API, and you have created multiple revisions of this API. If you add such an API to a product and try to publish it, you'll notice the following behaviour:

- The deployed revision of the API and the resources attached to it gets published.
- If you add a resource from an API which is not deployed, the same resource will not get published.
- If any resources attached to the product are present in the deployed API but not in its latest revision or draft, you should remove those resources before publishing the product.

When you create a product, you link it to one or more APIs. Also, the same API can be linked to multiple products. After you have linked an API to a product, all attributes of the API such as API resources and API documentation are implicitly part of the product.

A product is a vehicle that lets the application developer know which APIs are exposed on the . When you create an application, you select the product to include in the application. For each application that you create, generates an Application key and secret. Use this key to gain access to multiple products.

## 1.7.1 Create a Product

Explains how to create products to publish a bundle of API proxies together.

You create a product when you want to expose one or more API proxies to the Application Developer.

### Prerequisites

- You have the *APIPortal.Administrator* role assigned to you. For more information, see [User Roles in API Management](#).
- You've created the required API proxy on the *APIs* tab. For more information about how to create API proxies, see [Different Methods of Creating an API Proxy \[page 477\]](#).

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose *Engage*.
3. Go to the *Products* tab.

A list of published products appears.

You can view the number of calls made for all APIs in a product for the current month. The data is visible for each product in the *Calls* column and also on the details screen of the individual product.

You can click the refresh icon to get the latest data.

#### Note

- There may be a short delay before the data is refreshed.
- Number of calls won't be displayed for externally managed APIs.

The data is displayed according to metric specifications, for example:

- 999 shows as 999 and 1000 shows as 1K
- 999000 shows as 999K and 1000000 shows as 1M
- 1500000 shows as 1.5M and 1000000000 shows as 1G

4. To create a product, choose *Create*.

- For the product, enter a *Name* and *Title*, provide an introductory text in the *Short Text* field, and a brief description in the *Description* field.

#### Note

If you publish a product with a short introductory text, the short text appears on the corresponding API business hub enterprise pages. It appears on the product tiles on the API business hub enterprise landing page, and on the product details page. It also appears on the search result page for the searched products.

If you leave the short text field empty, the product description is displayed instead of the short text in the product tile.

- Specify the quota limits for this product.

#### Note

To enforce a quota on products, you must define verify API key and quota policies on the API. Setting quota limits on a product doesn't automatically enforce a quota on the API proxies. The quota set on the product takes precedence over that of the API proxy. It's a default limit that is referenced in quota policies that stipulate a uniform setting across all API proxies in the product. You can make runtime changes to the quota setting on an API product, and quota policies that reference the value automatically are updated with the new quota. For more information, see [Quota \[page 336\]](#).

You can use the sample payload given below to set **Verify API Key policy** for the required API:

#### Sample Code

```
<!--Specify in the APIKey element where to look for the variable
containing the api key-->
<VerifyAPIKey async='true' continueOnError='false' enabled='true'
xmlns='http://www.sap.com/apimgmt'>
 <APIKey ref='request.header.apikey' />
</VerifyAPIKey>
```

You can use the sample payload below on the same API to create **Quota policy**:

#### Sample Code

```
<!-- can be used to configure the number of request messages that an app
is allowed to submit to an API over a course of unit time -->
<Quota async="false" continueOnError="false" enabled="true"
type="calendar" xmlns="http://www.sap.com/apimgmt">
 <Identifier ref='verifyapikey.vap1.client_id' />
 <!-- specifies the number of requests allowed for the API
Proxy -->
 <Allow
countRef="verifyapikey.vap1.apiproduct.developer.quota.limit" count="100"/>
 <!-- the interval of time for which the quota should be
applied -->
 <Interval
ref="verifyapikey.vap1.apiproduct.developer.quota.interval">1</Interval>
 <!-- used to specify if a central counter should be
maintained and continuously synchronized across all message processors -->
 <Distributed>true</Distributed>
 <!-- Use to specify the date and time when the quota
counter will begin counting,
regardless of whether any requests have
been received from any apps -->
```

```

 <StartTime>2015-11-11 12:00:00</StartTime>
 <!-- if set to true, the distributed quota counter is
 updated synchronously. This means that
 the update to the counter will be made at
 the same time the API call is quota-checked -->
 <Synchronous>true</Synchronous>
 <!-- Use to specify the unit of time applicable to the
 quota. Can be second, minute, hour, day, or month -->
 <TimeUnit
 ref="verifyapikey.vapl.apiproduct.developer.quota.timeunit">month</
 TimeUnit>
 </Quota>

```

You'll notice that the same API key is used in the quota policy. You can now update the policy.

- To set the scope at product level to restrict the access of the authorization token for each application, specify the scope in the [Scope](#) field. For example, you can set the scope as Read-only, Read-write, and so on.

The OAuth 2.0 policy provides a way to limit the amount of access that is granted to an access token. For example, an access token issued to a client app may be granted READ and WRITE access to protected resources, or just READ access. You can implement your APIs to enforce any scope or combination of scopes you wish. So, if a client receives a token that has READ scope, and it tries to call an API endpoint that requires WRITE access, the call will fail.

Each product can have zero to many scopes assigned. These scopes can be assigned when the product is created or later. Scopes exist as a list of names and are included in the metadata associated with each product.

- In the [APIs](#) section, choose [Add](#).
- In the [Add APIs](#) window, select the required APIs and the corresponding resources.

### Note

While selecting APIs and its resources for product creation, the following behaviours apply when API calls are made to the selected API proxies and resources:

- Product creation in API Management provides precedence for product to path (resource) mapping over product to API mapping. Let's understand this behavior with an example:  
Let's say you want to create a product P1 that consists of 2 APIs namely API\_1 and API\_2.  
API\_1 contains resources namely R1 and R2. Whereas, API\_2 contains resources namely R2 and R3. That is API\_1=R1, R2 and API\_2=R2, R3  
As you can see, R2 is a common resource that exists in both the APIs.  
Now, for your product creation, let's say you select resources R1, R2 of API\_1 and resource R3 of API\_2. Thus, your product consists of resources R1 and R2 from API\_1 and R3 from API\_2. That is P1=R1, R2, R3.  
With the above resource selection criteria, API Management still allows API calls to be made to the resource R2 of API\_2 even though you had not explicitly selected the resource under API\_2 during product creation.
- If you want to publish a product with selective resource paths from multiple API proxies, you must ensure that the API proxies should have a common resource path.  
Consider the following example: You are trying to publish Product P1, with API proxy A1 having a resource path R1 and R2 and another API proxy A2. You must ensure that the API proxy A2 should have a common resource path as API proxy A1, which can be either R1 or R2.
- API Management doesn't support API calls to those resources whose path starts with a /\$ character. That is, if you create a product by attaching individual resources, then API calls to those resources whose path starts with /\$<resource\_name> don't work. However, when you attach the

whole API and none of its resources to a product, then API calls made to those resources of the selected API still works irrespective of whether the path starts with /\$ character or not.

#### Note

- If you attempt to add a resource to a product from an API that has not been deployed, the same resource will not be published.
- When publishing products, it's important to note that resources are available from the deployed API, rather than from the latest revision or draft of the API. Additionally, if a deployed resource is not available in the latest revision, you won't be able to attach that resource to the product.

#### Note

Select at least one API to publish a product.

#### Note

Make sure that the API is deployed before attaching it to a product. If you try to publish a product that has an API with saved changes attached to it, the following error message appears: "The API proxy attach to the product has some changes that aren't deployed yet."

Similarly, if the product has multiple API proxies attached to it, and few of the API proxies have changes that are saved but not deployed, you'll receive the following message when you try to publish the product: "The following API proxies attached to the product weren't published as they have changes that aren't yet deployed:"

#### 10. Choose *OK*.

The selected API proxies are listed on the *APIs* tab.

#### 11. Provide permissions to user roles to either discover or subscribe to the product.

#### 12. In the *Rate plans* section, choose *Add*.

#### 13. In the *Add Rate Plan* window, select the rate plans that you want to add to this product and choose *OK*.

#### 14. (Optional) In the *Custom Attribute* section, specify the custom attributes you want to add to the product. A custom attribute is a name/value pair, which can be used in multiple ways, including influencing the runtime behavior of an API proxy. Custom attributes provide the flexibility to extend the functionality based on attribute value, which can be set or read during the API proxy execution flow. These attributes can be accessed during an API call via the following policies: Verify API key, Access token, and Access entity.

#### Note

You can add a maximum of 18 custom attributes.

For example, you can create a custom attribute named `IsConfidential` with a value of Yes or No. Later, in your API proxy flow, you can check the value of the API product's `IsConfidential` attribute (for example, using the `verifyapikey.<policy_name>.apiproduct.IsConfidential` variable, which would be available automatically after you have created the custom attribute). If the value is Yes, you can throw an error, for example as shown below using the Raise Fault policy.

#### Sample Code

```
Warning: You do not have relevant authorizations to access the information.
```



## Note

The `verifyapikey.<policy_name>.apiproduct.IsConfidential` would be available only if the Verify API Key policy in your API proxy is executed successfully.

You can use the following sample payload to set Verify API Key policy for the required API:

## Sample Code

```
<!--Specify in the APIKey element where to look for the variable
containing the api key-->
<VerifyAPIKey async='true' continueOnError='false' enabled='true'
xmlns='http://www.sap.com/apimgmt'>
 <APIKey ref='request.queryparam.apikey' />
</VerifyAPIKey>
```

You can use the following condition and sample payload to set the Raise Fault policy:

## Sample Code

```
verifyapikey.Verify-API-Key.apiproduct.IsConfidential=True
```

## Sample Code

```
<!-- can be used to create custom messages in case of an error condition
-->
<RaiseFault async="true" continueOnError="false" enabled="true"
xmlns="http://www.sap.com/apimgmt">
 <!-- Defines the response message returned to the requesting client -->
 <FaultResponse>
 <Set>
 <!-- Sets or overwrites HTTP headers in the response message -->
 <Headers/>
 <Payload contentType="text/plain"> Warning: You do not have
relevant authorizations to access the information.
</Payload> <StatusCode>500</StatusCode>
 <!-- sets the reason phrase of the response -->
 <ReasonPhrase>Server Error</ReasonPhrase>
 </Set>
</FaultResponse>
<IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
</RaiseFault>
```

## Remember

- You can add externally managed APIs to a product but will not be able to add rate plans or add custom attributes for them.
- If your product contains only externally managed APIs, the user will not be able to subscribe to the the product.

15. Once you have filled in all the required details for the product, you can choose one of the following two actions:

- **Save as Draft** - The resulting state of the product is Draft, and you can publish the product anytime. The product cannot be used for creating applications when it is in the Draft state.
- **Publish** - The resulting state of the product is Published, and is available for creating applications. You can edit the published product anytime.

### Note

When you're selectively publishing the resources of an API proxy associated with a product, please make sure that the API resources shouldn't contain any special characters. For example, you can use `/Employee` and not `/Employee('{test}')` as it contains special characters.

16. If you want to delete a product, select the required product from the catalog and choose *Delete*. if it is being used in an application.
17. If you want to edit a product, select the required product, in the details view select *Edit*.

### Note

Alternatively, you can use the following service to update the product:

#### Sample Code

sample payload to update a product

```
--batch_1ddf-343f-3338
Content-Type: multipart/mixed; boundary=changeset_418f-1c1d-b76b
--changeset_418f-1c1d-b76b
Content-Type: application/http
Content-Transfer-Encoding: binary
PUT APIProducts(name='<Product_Name>') HTTP/1.1
x-csrf-token: E2BE219B16E5608F21EE5A7765E1318C
Accept-Language: en-US
Accept: application/json
MaxDataServiceVersion: 2.0
DataServiceVersion: 2.0
Content-Type: application/json
Content-Length: <Length of below payload>
{"name": "<Product_Name>", "title": "<Product_Title>", "scope": "", "description": "<Product_Description>", "version": "1", "status_code": "PUBLISHED", "isRestricted": <true or false>, "isPublished": true, "quotaCount": -99, "quotaInterval": -99, "quotaTimeUnit": null}
--changeset_418f-1c1d-b76b--
--batch_1ddf-343f-3338--
```

## 1.7.1.1 Assign Permission to a Product via UI

Assign permission to a product in the .

### Prerequisites

- You are assigned the `APIPortal.Administrator` role.
- You must have created a custom role on the SAP BTP Cockpit Cloud Foundry environment. For more information on creating a custom role, refer [here \[page 160\]](#).

## Context

Whenever you create a product or edit a draft product, permissions can be added to product. Use this procedure to grant permission to user roles for discovering and subscribing to the product in the API business hub enterprise. Only users who are assigned the required role can discover and subscribe to the product.

### Note

Currently, we do not support assigning of permissions to the products defined in the remote API portals that are connected to a centralised API business hub enterprise.

\*Remote API portals are those that are not in the same subaccount as the centralised API business hub enterprise and are configured via the manage connections. For more information, [Centralized API business hub enterprise \[New Design\] \[page 186\]](#) and [Centralized API business hub enterprise \[Classic Design\] \[page 177\]](#).

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose *Engage*.
3. Go to the *Products* tab and choose *Permission*.

Here, whenever a product is created or a draft product is edited, permissions can be added to product.

4. Select a role from the *Discovery* dropdown list. Only users who are assigned the selected role can discover this product in the API business hub enterprise.
5. Select a role from the *Subscription* dropdown list. Only users who are assigned the selected role can subscribe to this product in the API business hub enterprise.
  - You can change the roles selected for *Discovery* and *Subscription* by choosing *Edit*.
  - You can remove the roles selected for *Discovery* and *Subscription* by choosing *Remove Role*.

## Related Information

[Create a Product \[page 653\]](#)

## 1.7.2 View Applications

In the context of API Management, an application is the unit of API consumption.

### Context

API Management enables the creation of secure API proxies for your APIs. These proxies are protected using "Appkey" and "Secret". Application developers are required to acquire these credentials in order to utilize the API proxies exposed through the various products. They need to declare the usage of these products by creating an application from API business hub enterprise.

As an admin, you can view the list of applications that the developers have created in the API business hub enterprise, along with the developer details, associated products and AppKey and secret.

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose *Engage*.
3. Navigate to the *Applications* tab page.

A list of Applications appears. If you have logged on to the with the role APIPortal.Administrator, then you can view the Application key and secret.

You can view the number of calls made for all APIs in an application for the current month. The data is visible for each application in the *Calls* column and also in the details screen of individual application.

You can click on the refresh icon to obtain the latest data.

#### Note

There is some delay in reflecting the latest data.

Notion used to display the data is as per metric specifications, for example:

- 999 shows as 999 and 1000 shows as 1k
- 999000 shows as 999K and 1000000 shows as 1M
- 1500000 shows as 1.5M and 1000000000 shows as 1G

### Related Information

[Create an Application \[Classic Design\] \[page 686\]](#)

## 1.8 Consume API Proxies

Consume API proxies via the API business hub enterprise. In the API business hub enterprise, an application developer registers, explores the API exposed by customers, creates applications, and tests API proxies.

### ⚠ Caution

Effective June 2024, the classic design of the API business hub enterprise will be deprecated and will no longer be accessible. The new design of the API business hub enterprise will be set as your default design from March 2024. For more information, see [Configure the API business hub enterprise \[New Design\] \[page 675\]](#).

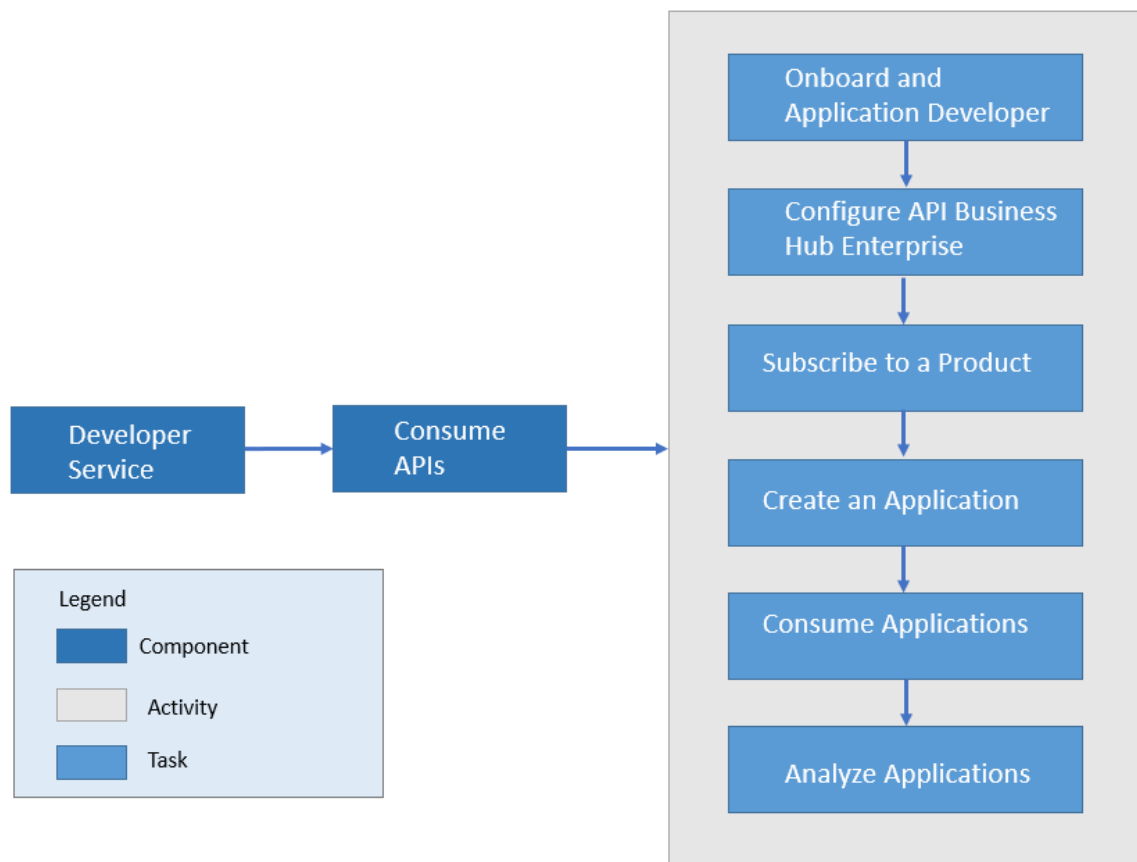
If you've added API business hub enterprise as a capability with Integration suite, or if you've subscribed to the API business hub enterprise as part of the standalone API Management subscription, you have the option to experience the new design of the API business hub enterprise user interface along with the classic design.

When you login for the very first time, you'll still see the classic design of the API business hub enterprise. However, you can use the toggle switch to view the new design. At this point of time, only the Site Administrator can use the toggle switch.

### ℹ Note

By default, the Site Administrator has an option to switch from classic to new design and set the new design as the default UI using the **Site Editor**. The Site Administrator has the right to enable the configuration to let all the other users switch between the old and the new design. For more information, see [Customize the Visual Format of the API business hub enterprise \[page 677\]](#).

Refer this video for a complete walk through of the new design of the API business hub enterprise:



API business hub enterprise is an application that provides a common platform for Application developers to consume API proxies. Every API Management customer is provided with their own API business hub enterprise application on cloud. The API business hub enterprise offers capabilities to onboard application developers, explore and test API proxies, create and subscribe to Applications.

The API business hub enterprise supports the following features:

- **Onboard an Application developer**- To explore the API proxies and subscribe to an Application, an Application developer must be registered to the API business hub enterprise. On registering, the Application developer is provided access to the API business hub enterprise.
- **Browse Catalog**- Explore the Products (assembled APIs) available in the Catalog store, navigate to individual API proxies, read the API Documentation, and view the resources attached to the API proxies.
- **Create Applications** – An Application developer can create on or more applications to consume API proxies. To consume the API proxies, an Application developer must subscribe to an Application (assembled Products). It is by subscribing to an Application that you return to the developer the key required to access the API proxies.
- **Download JSON**- You can download the open API specification for the APIs that are part of the API business hub enterprise in JSON format. This enables the developer to use the metadata of the APIs for various aspects such as code/SDK generation for developing applications.
- **Download SDK**- You can also download the client software development kit (SDK) for developers through a non-commercial license on open source sites. You can use this SDK for developing applications.
- **Test API Proxies** - You can test the API proxies and understand the runtime behavior of the API proxies better. Use the Test Console to explore the resources associated with an API and execute the operations.

## Related Information

[Configure the API business hub enterprise \[Classic Design\] \[page 671\]](#)

[Configure the API business hub enterprise \[New Design\] \[page 675\]](#)

### 1.8.1 Onboard an Application Developer

Explains how API administrators can onboard application developers so they can access the API business hub enterprise.

#### Context

A user must be onboarded to API business hub enterprise only via Self-registration or **Add User** flow.

To provide application developers with access to the API business hub enterprise, the API Administrator first has to onboard them. The steps to onboard an application developer are as follows:

#### Procedure

1. The application developers log on to the API business hub enterprise application with their IDP user credentials, and register to the API business hub enterprise. For more information, see [Register on API business hub enterprise \[page 664\]](#).
2. The API administrator approves or rejects the request to access the API business hub enterprise. For more information, see [Managing the Access Request of the Users \[Classic Design\] \[page 665\]](#).

If you haven't enabled the automatic creation of shadow users, and you've not explicitly created shadow users for your developers, then they're unable to log on to the application, and they're asked to contact the administrator. For more information, see [Shadow Users \[page 176\]](#)

## 1.8.1.1 Register on API business hub enterprise

Procedure to register as an application developer on the API business hub enterprise to view the products available in the catalog store. The API business hub enterprise also enables you to explore the APIs, read the associated API documentation, and view resources.

### Prerequisites

- As a developer you're trying to self-register:
  - You're already a valid Application IDP user.
  - The admin has already added your email ID in the subaccount.

#### Note

If the *AuthGroup.API.ApplicationDeveloper* role is already assigned to you by the SAP BTP admin or via the IDP Role Collection mapping, you will get automatically registered as an application developer in API business hub enterprise when you logon for the first time.

If you don't have the *AuthGroup.API.ApplicationDeveloper* role assigned to you in SAP BTP cockpit, complete the self-registration process to access all the functionalities and features of API business hub enterprise.

Please note that the *AuthGroup.API.ApplicationDeveloper* role that has been assigned to you will only take effect if you login to API business hub enterprise. Only relevant for New Design of the API business hub enterprise.

- As an Admin you're trying to onboard multiple users:
  - The admin has already added the email IDs of the users in the subaccount.
  - The admin has assigned the *AuthGroup.API.Admin* role to all the users.

#### Note

While onboarding multiple users, it is recommended that you don't assign the *AuthGroup.API.Admin* role to all the users as this will enable the developers to take on the admin role. Instead you can automate the process of onboarding multiple users by using the API "[API Business Hub Enterprise - Registering Users\(CF\)](#)".

In this case, admin approval is not required. When the user logs in and chooses the *Register* button, they get auto registered as developers.

#### Note

Consider the following behavior for auto-registration:

- **Use Case 1: User is no longer in the organization:**

If the *AuthGroup.API.ApplicationDeveloper* role is removed from either the SAP BTP cockpit or the IDP Role Collection mapping, as an admin, you should also ensure that the *AuthGroup.API.ApplicationDeveloper* role is removed from the API business hub enterprise, or vice versa. Failing to do so may lead to confusion and discrepancies.
- **Use Case 2: User is still in the organization:**

If a user is still part of the organization, the role removal in BTP will take effect in the API business hub enterprise once the user logs in. If this user wishes to access the API business hub enterprise,



they must either follow the self-registration process or have this role assigned to them from the SAP BTP Cockpit.

## Context

The procedure below describes the sequence of steps when as a developer you're trying to self-register:

## Procedure

1. Log on to the API business hub enterprise application with your IDP user credentials.
2. To register to the API business hub enterprise as an Application developer, choose [Register](#).  
A dialog box with the prepopulated data such as, your first name, last name, and e-mail address appears.
3. Enter the country/region and reason for requesting access to the API business hub enterprise.
4. Choose [OK](#).

The request is sent to the administrator with the `AuthGroup.API.Admin` role.

- If the administrator approves your request, you'll receive an e-mail notification. You can log in to the API business hub enterprise via the link provided in the e-mail.
- If the administrator rejects the request, you'll receive an e-mail notification with the reason for the rejection. When you log on to the application, you'll see the reason for request rejection on the display page.

### Note

Application Developers can now email to the administrator by replying to the email notification they receive for any queries regarding their access request to the API business hub enterprise application.

## 1.8.1.2 Managing the Access Request of the Users [Classic Design]

Procedure to provide or reject access to an Application developer for using the API business hub enterprise.

## Prerequisites

You're assigned the `AuthGroup.API.Admin` role.

## Note

This document describes the classic design of the API business hub enterprise. To view the documentation for the new design, see [Managing the Access Request of the Users \[New Design\] \[page 667\]](#).

## Procedure

1. Log on to the API business hub enterprise.
2. Choose **Manage > Manage Users**.

Use the *Manage Users* page to onboard the application developers. For assigning roles to other users, use the SAP BTP Cockpit.

3. Add the email id in the textbox.

This email id is used to receive email notification of the pending requests and send approvals.

4. To view the pending requests, navigate to *Pending* section.
5. In the pending section, **Approve** or **Reject** the request by choosing the corresponding action item in the *Actions* column.

On accepting the request, an approval email is sent to the requester. On rejecting a request, you need to provide a reason and an email notification is sent to the requester with the mentioned reason.

To view registered users, navigate to *Registered Users* section.

In the *Registered Users* section, you can perform the following:

- Edit an existing user to add or remove user roles.
- Register a new user by selecting the **+ Add User** icon.
  - In the *Add User* dialog:
    1. Enter the *User ID* and the user details like *First Name*, *Last Name* and *Email ID*.
    2. Under *Assigned Roles*, identify the needed scope and select the roles accordingly:

Roles	Description
Administrator	Manages user registration. Create and delete applications on behalf of application developers. In addition, create custom attributes and import the app key.
Developer	Create applications and check billing and metering data. Test APIs and view analytics data.
Site Administrator	Configure updates, and perform portal changes like uploading the logo, changing the name and the description, and changing the footer links for the site.

Roles	Description
Content Administrator	Manages content categories.

## Related Information

[Configure the API business hub enterprise \[Classic Design\] \[page 671\]](#)

[Revoke Access \[Classic Design\] \[page 668\]](#)

### 1.8.1.3 Managing the Access Request of the Users [New Design]

As an API administrator, you can approve or reject the access request made by an application developer to use the API business hub enterprise.

## Prerequisites

You're assigned the *AuthGroup.API.Admin* role.

#### Note

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Managing the Access Request of the Users \[Classic Design\] \[page 665\]](#).

## Procedure

1. Log on to the API business hub enterprise.

Use the *Manage Users* page to approve or reject the developer's registration requests and manage the roles of the registered users. For assigning roles to the users, use the SAP BTP Cockpit.

2. Choose **Enterprise Manager** > **Manage Users** > **E-mail Configuration** and add the administrator's email in the *E-mail Configuration* textbox.

The administrator receives email notification of the pending developer registration requests on this email id. Also, the e-mail notifications to the developers or users are sent from this e-mail id.

#### Note

Only one administrator's email address can be entered in the *E-mail Configuration* textbox.

- To view the pending requests, navigate to [Manage Users](#) > [New Requests](#).
- Look for the request and choose [Accept Request](#) from the [Actions](#) column. The application developer can now access the API business hub enterprise.

If you don't wish to provide access to the user, choose [Decline Request](#) from the [Actions](#) column.

On accepting the request, an approval email is sent to the requester. On rejecting a request, you need to provide a reason for rejection; an email notification is sent to the requester.

You can also view the registered users by choosing [Manage Connections](#) > [Registered Users](#).

On the [Registered Users](#) page, you can:

- Register a new user by choosing [Add User](#).

#### Note

A user must be onboarded to API business hub enterprise only via Self-registration or **Add User** flow.

In the [Add User](#) dialog:

- Enter the [User ID](#) and the user details like [First Name](#), [Last Name](#) and [Email ID](#).
- Under [Assigned Roles](#), identify the needed scope and select the roles accordingly:

Roles	Description
Administrator	Manages user registration. Create and delete applications on behalf of application developers. In addition, create custom attributes and import the app key.
Developer	Create applications and check billing and metering data. Test APIs and view analytics data.
Site Administrator	Configure updates, and perform portal changes like uploading the logo, changing the name and the description, and changing the footer links for the site.
Content Administrator	Manages content categories.

- Edit an existing user to add or remove the [Assigned Roles](#).

## 1.8.1.4 Revoke Access [Classic Design]

Revoke the access of an application developer.

### Prerequisites

You are an API administrator and the role [AuthGroup.API.Admin](#) is assigned to your user.

## Context

As an API administrator, you use this procedure to revoke an application developer's access for using the API business hub enterprise.

### Note

This document describes the classic design of the API business hub enterprise. To view the documentation for the new design, see [Revoke Access \[New Design\]](#) [page 669].

## Procedure

1. Log on to the API business hub enterprise.
2. Choose ► [Manage](#) ► [Manage Users](#) ▾.
3. Go to the [Registered](#) section. From the list of application developers, select the application developer whose access you want to revoke and choose the revoke user action item under the [Actions](#) column.
4. In the [Revoke](#) window, provide a reason for revoking the access.

### Note

By revoking roles, user will lose all the roles assigned. However, user account will be retained.

## Results

You have revoked the access of the user from using API business hub enterprise successfully.

### 1.8.1.5 Revoke Access [New Design]

Revoke the access of an application developer.

## Prerequisites

You are an API administrator and the role [AuthGroup.API.Admin](#) is assigned to your user.


## Context

As an API administrator, you use this procedure to revoke an application developer's access for using the API business hub enterprise.

### Note

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Revoke Access \[Classic Design\] \[page 668\]](#).

## Procedure

1. Log on to the API business hub enterprise.
2. Choose ► [Enterprise Manager](#) ► [Manage Users](#) ► [Registered Users](#) ►.
3. From the list of application developers, select the application developer whose access you want to revoke and choose the  [Revoke User](#) icon under the *Actions* column.
4. In the [Revoke](#) window, provide a reason for revoking the access.

### Note

By revoking roles, users lose all the roles assigned to them. However, user account will be retained.

## Results

You have revoked the access of the user from using API business hub enterprise successfully.

### 1.8.1.6 Delete Data of Unregistered Users

SAP API Management stores the data of users who have logged on to the developer portal but have not registered. This topic describes the service used to delete the data of such users.

## Prerequisites

You are assigned the AuthGroup.API.Admin role.

## Context

## Procedure

Run the following service using the standard REST console:

- Service URL: `https://<Dev-Portal-URL>/api/1.0/offboarding/{userId}`
- Method: POST
- Request Header: `x-csrf-token: fetch`
- Content Type: `application/json`
- Response: 201

The user data is deleted.

## 1.8.2 Configure the API business hub enterprise [Classic Design]

You can configure the API business hub enterprise to personalize it for your organization.

### ⚠ Caution

Effective June 2024, the classic design of the API business hub enterprise will be deprecated and will no longer be accessible. The new design of the API business hub enterprise will be set as your default design from March 2024. For more information, see [Configure the API business hub enterprise \[New Design\] \[page 675\]](#).

### 📘 Note

This document describes the classic design of the API business hub enterprise. To view the documentation for the new design, see [Configure the API business hub enterprise \[New Design\] \[page 675\]](#).

Select *Manage* from the banner. Modify the following sections to personalize the API business hub enterprise.

The prerequisite varies for each section. Sections are visible to the user based on the role assigned to the user. For example, *General* section is visible to only users assigned with site admin role. For more information on API business hub enterprise user roles, see [Assigning User Roles](#).

### General

Prerequisite: You are assigned the site admin role.

Section	Description
Company Logo	Upload an image file for your logo, and save the changes.

Section	Description
Color Scheme	<p>Information about the color scheme used.</p> <p>When you upload a logo, the dark color of the navigation area at the top of the screens is replaced with white, since many logos require a white background.</p> <p>Other colors on the Web site are changed to neutral tones to avoid visual conflicts with your company's logo colors.</p>

## Home Page

Prerequisite:

- You are assigned the *AuthGroup.Site.Admin* role to view the following sections: *Name and Description* and *Updates*.
- You are assigned the content admin role to view *Navigation Categories*.

From the homepage, you can view and search the categories, APIs, and products.

Section	Description
Name and Description	Edit the default name and description for your application.
Updates	Configure updates to be displayed on the Home page. For more information on how to add, edit, or delete an update, see <a href="#">Manage Updates [Classic Design] [page 673]</a> .
Navigation Categories	Configure navigation categories to be displayed on the Home page. For more information on how to add, edit, or delete a category, see <a href="#">Manage Navigation Categories [Classic Design] [page 674]</a> .

## Manage Users

Prerequisite: You are assigned the admin role.

Section	Description
E-mail Configuration	Provide the administrators e-mail id.
Pending Requests	Information about pending user requests. You can either accept or reject the requests. For more information, see <a href="#">Managing the Access Request of the Users [Classic Design] [page 665]</a> .
Registered Users	Information about users registered. You can either edit the roles for an existing user or register a new user. For more information, see <a href="#">Managing the Access Request of the Users [Classic Design] [page 665]</a> .

## Manage API Portal Connection

Prerequisite: You're assigned the *AuthGroup.API.Admin* role.



As an API business hub enterprise administrator, you can approve or reject the connection requests from this page. You can also view the requests that you have previously approved or rejected.

### Reference Links

Prerequisite: You are assigned the site admin role.

You can add, edit, and customize the links that appear at the bottom of the page here. Links are grouped into sets of three. Click **+** to add and edit Web site updates and news. In the *Add Link* dialog window, enter the link title and URL, and choose *Save*.

## 1.8.2.1 Manage Updates [Classic Design]

Configure the updates to be displayed in the *Updates* section on the home page.

### Prerequisites

You have the site admin role assigned to you.

### Context

Use the following procedure to configure updates.

### Procedure

1. Log on to the API business hub enterprise, and navigate to ► *Manage* ► *Home Page* ► *Updates* ► *Configure Updates* ►.
2. To add an update, click *Add Update* icon. In the *Add Update* screen that opens, enter the following details:

Name	Description
<b>Title</b>	Provide a title for the update.
<b>Description</b>	Provide a description for the update.
<b>Link</b>	Provide the details of a reference link. Link is an optional field to provide more information on the update.

3. Save the changes.

Newly configured update is visible in the *Configured Updates* section. In this section, you can perform the following:

- Reorder the updates by using the *Move Up* and *Move Down* action icons.

- Edit an update by choosing the edit action icon.
- Delete an update by choosing the delete action icon.

## 1.8.2.2 Manage Navigation Categories [Classic Design]

Navigation categories are displayed on the home page.

### Prerequisites

You have the content admin role assigned to you.

### Context

#### Note

This document describes the classic design of the API business hub enterprise. To view the documentation for the new design, see [Manage Domain Categories \[New Design\] \[page 679\]](#).

Use the following procedure to configure navigation categories.

### Procedure

1. Log on to the API business hub enterprise, and navigate to [Manage](#) > [Home Page](#) > [Navigation Categories](#).
2. To add a category, choose [Add Category](#) icon. In the [Add Category](#) screen that opens, enter the following details:

Name	Description
<b>Category Name</b>	Provide a name for the category.
<b>Category Title</b>	Provide a title for the category. Categories are identified by their title on the home screen.
<b>Description</b>	Provide a description for the category.

3. To add products, choose [Add Products](#) button. In the [Add Products](#) window that opens, select the products that you want to add to this category.
4. Save the changes.

Newly configured category is visible in the [Navigation Categories](#) section. In this section, you can perform the following:

- Reorder the updates by using the *Move Up* and *Move Down* action icons.
- Edit a category by choosing the edit action icon.
- Delete a category by choosing the delete action icon.

## 1.8.3 Configure the API business hub enterprise [New Design]

You can configure the API business hub enterprise to personalize it for your organization.

### Note

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Configure the API business hub enterprise \[Classic Design\] \[page 671\]](#).

Modify the following to personalize the API business hub enterprise:

You can use the...	To...	Role that you must be assigned to...	For more information, see...
<a href="#">Home Page</a>	View and search the categories, APIs, and products	AuthGroup.API.Admin AuthGroup.API.Application-Developer	<a href="#">Register on API business hub enterprise [page 664]</a>
		<div style="border: 1px solid #ccc; padding: 5px;"> <p><b>Note</b></p> <p>The <i>AuthGroup.API.ApplicationDeveloper</i> role is assigned by default to a user who onboards to the API business hub enterprise using the Self-registration process or via <i>Add User</i> flow.</p> </div>	
<a href="#">Site Editor</a>	Customize the visual layout of the API business hub enterprise	<ul style="list-style-type: none"> <li>AuthGroup.Site.Admin</li> <li>AuthGroup.API.Admin</li> </ul>	<a href="#">Customize the Visual Format of the API business hub enterprise [page 677]</a>
<ul style="list-style-type: none"> <li>▶ <a href="#">Enterprise Manager</a></li> <li>▶ <a href="#">Manage API Management Connections</a> ▶</li> </ul>	Approve and reject the pending connection requests and update the API portal access credentials	<ul style="list-style-type: none"> <li>AuthGroup.APIPortalRegistration</li> </ul>	<a href="#">Approve the Pending Connection Requests [New Design] [page 192]</a>

You can use the...	To...	Role that you must be assigned to...	For more information, see...
<a href="#">Enterprise Manager</a> <a href="#">Manage Users</a>	Add and Revoke user access to the API Business Hub Enterprise	AuthGroup.API.Admin	<a href="#">Managing the Access Request of the Users [New Design] [page 667]</a>  <a href="#">Revoke Access [New Design] [page 669]</a>
<a href="#">Enterprise Manager</a> <a href="#">Manage Domain Categories</a>	Create and edit domain specific categories.	<ul style="list-style-type: none"> <li>AuthGroup.Content.Admin</li> <li>AuthGroup.API.Admin</li> </ul>	<a href="#">Manage Domain Categories [New Design] [page 679]</a>  Also, to add the <i>AuthGroup.Content.Admin</i> role, see the table in <a href="#">Manage Domain Categories [New Design] [page 679]</a> .
<a href="#">Enterprise Manager</a> <a href="#">Manage Notifications</a>	Configure notifications for providing information to the API business hub enterprise end users on any website updates or news items.	<ul style="list-style-type: none"> <li>AuthGroup.Site.Admin</li> </ul>	<a href="#">Manage Notifications [New Design] [page 681]</a>
<a href="#">My Workspace</a>	Create applications and view your applications, costs and analyze reports.	AuthGroup.API.Admin	<a href="#">Create an Application [New Design] [page 693]</a>

**Note**  
Administrators can create applications on behalf of a developer and can see all the applications across developers.

**Note**  
AuthGroup.API.ApplicationDeveloper

**Note**  
Application developers can create applications and see only the applications created by him.

You can use the...	To...	Role that you must be assigned to...	For more information, see...
<a href="#">Test Environment</a>	Test the runtime behaviour of APIs	AuthGroup.API.Application-Developer	<a href="#">Test Runtime Behavior of APIs [New Design] [page 699]</a>

**Note**

Application developers can test the runtime behaviour of APIs.

## 1.8.4 Customize the Visual Format of the API business hub enterprise

As a Site Administrator, you can customize the visual layout of the API business hub enterprise using the Site Editor. The customizations you make using the Site Editor appear to the other users in the system.

### Prerequisites

You already have the [Site Administrator](#) role assigned to you.

### Site Editor

You can use the [Site Editor](#) to:

- Edit site name, logo, and description
- Modify header design
- Set default design for the site
- Change banner image
- Customize background settings
- Customize footer

### Edit Site Name and Logo

To modify the site name and logo,

1. Choose the [Edit Site Name and Logo](#) tab.

2. Enter the name in the *Site Name* field.
3. Upload or drag and drop an image for the logo in the *Site Logo* field.

## Edit Header Design

To customize the header design,

1. Choose the *Edit Header Design* tab.
2. In the *Edit Header Design* popup, choose colors for the header bar, text, and the selection bar.

## Set Default Design

To set a default design,

1. Choose the *Set Default Design* tab.
2. Select the *Set current design as default design* checkbox.
3. To allow users to toggle between the old and the new interface, select the *Allow users to toggle between old design and new design* checkbox.

### Note

Once the new design is activated, the changes are permanent, and you can't revert to the classic design.

## Change Banner Image

To change the banner image,

1. Choose the *Change Banner Image* tab.
2. You can upload the banner image in the *Upload Image* field or select from the predefined library available.

## Edit Description

Choose the *Edit Description* tab to modify the *Title* and *Subtitle* of the home page.

### Note

As a part of banner description, you can use the markdown language to add links and email addresses to the *Title* and *Subtitle* fields within the *Edit Description* dialog. For example, [SAP]([www.sap.com](http://www.sap.com)) and <mailto:john.doe@example.com>.

You can also add multiple lines in the sub-titles using markdown. If you want to move to a new line, end the previous line with a backslash (\).

## Edit Banner Settings

To modify the banner settings,

1. Choose the [Edit Banner Settings](#) tab.
2. You can adjust the height of the banner image by using the [Minimum Height of the Background Image](#) field.
3. Colors for the text and search box can be selected using the [Text Color](#) and [Search Box Color](#) fields.

## Customize Footer

1. To add reference links to the footer, choose the [Edit Footer](#) tab.
  - Choose [Add a Link](#) to create a new reference link.
  - To bundle relevant links into a group, choose [Group links](#).
2. Choose the [Edit Footer Design](#) tab to customize the footer bar color, text color, and hover link text color.

## Publish Changes

The customizations you've made to the API business hub enterprise layout will be available to the users once you publish the changes.

## 1.8.5 Manage Domain Categories [New Design]

Domain categories are displayed on the API business hub enterprise home page.

### Prerequisites

You need the following roles to create and update categories:

- [AuthGroup.Content.Admin](#)  
To assign the role, see [Managing the Access Request of the Users \[New Design\] \[page 667\]](#).
- [AuthGroup.API.Admin](#)  
To assign the role, see .

### Context

Content administrators can use [Manage Content](#) to create domain categories and add the related products into relevant categories. They can also configure the order in which these categories and the contained products get displayed in the home page.

## Note

If you've configured the API business hub enterprise to connect to multiple API portals then you can add products from different API portals under one category. Whereas in classic design, you can only add products from one API portal under a category.

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Manage Navigation Categories \[Classic Design\] \[page 674\]](#).

Use the following procedure to configure navigation categories.

## Procedure

1. Log on to the API business hub enterprise.
2. Choose **Enterprise Manager** > **Manage Domain Categories** from the top navigation bar.
3. To add a category, choose **Add New Domain Category**.
4. Enter the following details in the **Add Domain Category** dialog and choose **Save**:

Name	Description
<b>Category Name</b>	Provide a name for the category.
<b>Category Title</b>	Provide a title for the category. Categories are identified by their title on the home screen.
<b>Description</b>	Provide a description for the category.

5. To add products, choose **+ Add Products** from the top-right corner of the newly created category pane. In the **Add Products** dialog, select the products that you want to add to this category and choose **Save**.
6. Save the changes.

Newly configured category is visible on the **Manage Content** page. For individual categories, you can perform the following:

- Reorder the categories by using the **^ Move Up** and **∨ Move Down** action icons.
- Edit a category by choosing the **edit** icon.
- Delete a category by choosing the **trash** icon.



## 1.8.6 Manage Notifications [New Design]

As a site administrator you can configure notifications for providing information to the API business hub enterprise end users on any website updates, events or news items.

### Prerequisites

You're assigned the *AuthGroup.Site.Admin* role. To assign the role, see [Managing the Access Request of the Users \[New Design\] \[page 667\]](#).

### Procedure


1. Log on to the API business hub enterprise.
2. Choose **► Enterprise Manager ► Notifications ►** from the top navigation bar.
3. Choose *Add Notification*.
4. Provide the following details on the *Add Notification* dialog:

<i>Topic Name</i>	Enter a name for the notification entity. <b>Example:</b> Experience the new design of the API business hub enterprise!
<i>Description</i>	Enter a description for the notification entity. <b>Example:</b> If you've subscribed to API business hub enterprise as part of Integration Suite subscription , we now have a new design of the user interface for you to experience.

5. If you want to provide more information through a link, for example, a link to a blog post, or a help document, choose *Add Link* and enter the link text and the URL.

#### Note

The *Topic Name* and the *Display Name* (link text ) support a maximum of 250 and 512 characters, respectively. The *Description* and the *URL* fields support 2048 characters each.

6. Choose *Add* to publish the notification.
7. After adding the notifications, enable the notification feature using the toggle switch on the *Manage Notifications* page. The notifications will be visible to the users under the  *notification bell* icon in the header only if the site administrator has explicitly enabled the notification.

## Results

The notification appears on the [Manage Notifications](#) page.

### Note

You can also, edit notifications, change the order of the notifications, and delete the notifications per your requirement.

## 1.8.7 Manage External Content (New Design)

On this page, you have the option to adjust the visibility of the Graph navigator on the API business hub enterprise.

### Prerequisites

- Assign the [AuthGroup.API.Admin](#) role.  
To assign the role, see .
- Enable [Graph](#) in Integration Suite. For more information, see [Activating and Managing Capabilities](#) and [Configuring User Access](#).

### Context

If the graph feature is activated in the Integration Suite, the [Configure Graph](#) setting will be enabled by default in [Manage External Content](#), and the [Graph](#) option will be displayed in the header. If you want to disable it, you can do so from this page.

If the graph feature is deactivated in the Integration Suite, the [Configure Graph](#) setting in [Manage External Content](#) will be disabled.

### Procedure

1. Log on to the API business hub enterprise.
2. Choose [Enterprise Manager](#) [Manage External Content](#) from the top navigation bar.
3. Use the slider button to enable/disable the graph feature.  
Choose [Yes/No](#) in the confirmation dialog.

## 1.8.8 Manage Access

As an API business hub enterprise admin, you have the authority to control the level of access for your users, allowing them to search, discover, and access the content available on the API business hub enterprise.

### Prerequisites

You need the following role to configure the access control checks:

- [AuthGroup.API.Admin](#)  
To assign this role, see [Assigning Role Collections to Users \[page 160\]](#) .

#### Note

The **Manage Access** feature is available only in the new design of the API business hub enterprise on the Cloud Foundry environment.

### Context

In API business hub enterprise, managing access for different users is important for several reasons like improving user productivity, resource optimization, privacy, and security.

User productivity is enhanced by granting users the appropriate access to carry out their tasks efficiently, without being overwhelmed by unnecessary information or resources. In this context, the **All Visitors** option allows anyone, whether logged in or not, to utilize the APIs without requiring authentication. However, the ability to consume the APIs still depends on obtaining the necessary developer role.

Moreover, by managing access, you can provide access to **Authenticated Users** who do not have a designated role, allowing them to access different pages of the API business hub enterprise based on their specific needs. This facilitates broader exploration and enables users to familiarize themselves with the available resources. Nevertheless, the ability to consume the APIs still relies on obtaining the necessary developer role.

To maintain privacy and security, you can grant access to **Authorized Users** who are logged in and possess the required developer role. This ensures that only authorized individuals can seamlessly access and consume the APIs while upholding privacy and security measures.

#### Note

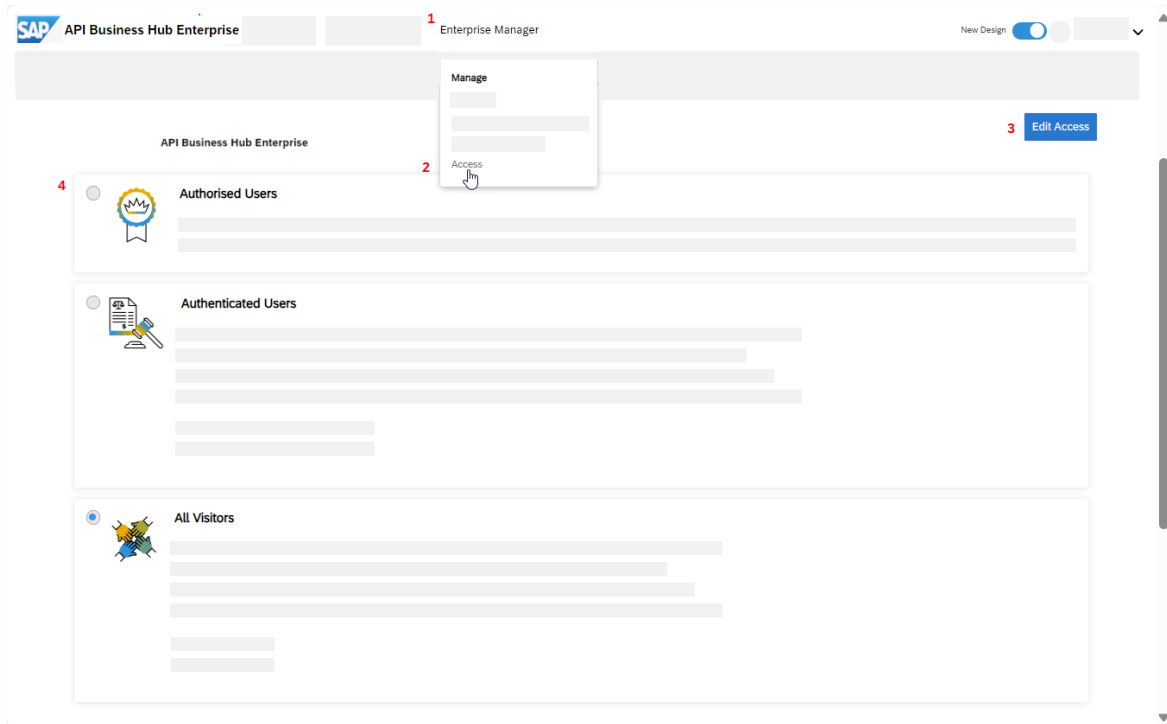
Access to the API business hub enterprise content using the API access plan is not affected by these permissions.

#### Note

As an administrator of API business hub enterprise, please note that when you update these permissions, it may take up to 5 minutes for the changes to be applied for other users of the API business hub enterprise.

## Procedure

1. Log on to the API business hub enterprise.
2. Choose ► [Enterprise Manager](#) ► [Manage Access](#) from the top navigation bar.



3. Choose [Edit Access](#) to manage the level of access for different kinds of users.

Select the appropriate permission from the following list:

- **Authorised Users:** These are the users who are registered as application developers or have the privilege of the application developer assigned through the SAP BTP cockpit. They can search, discover, and consume the contents of this API business hub enterprise application.

### Note

By default, the option for **Authorised Users** is selected. However, you have the flexibility to change it to either **All Visitors** or **Authenticated Users**, depending on your specific requirements.

- **Authenticated Users:** These users are not registered as application developers but are successfully authenticated by the API business hub enterprise application and can access the following pages:

- Home
- Search results
- Product Details

Optionally, you can select the **API Details** checkbox to provide access to the **API Details** page as well.

- **All Visitors:** These are the users who can visit the following pages even before they are authenticated by the API business hub enterprise application, that is without logging in to the API business hub enterprise application:

- Home
- Search results

- Product Details

Optionally, you can select the [API Details](#) checkbox to provide access to the [API Details](#) page as well.

4. Choose [Save](#) after providing the required permission.

## 1.8.9 Subscribe to a Product [Classic Design]

You can subscribe to a product and add it to an existing application or create a new application.

### Context

On the homepage, you can find the list of products. You can also view the various resources that each product has to offer.

#### ⚠ Caution

Effective June 2024, the classic design of the API business hub enterprise will be deprecated and will no longer be accessible. The new design of the API business hub enterprise will be set as your default design from March 2024. For more information, see [Configure the API business hub enterprise \[New Design\] \[page 675\]](#).

#### ℹ Note

This document describes the classic design of the API business hub enterprise. To view the documentation for the new design, see [Subscribe to a Product \[New Design\] \[page 686\]](#).

### Procedure

1. Log on to the API business hub enterprise.
2. You can either look for the product on the homepage, or use the search bar to search for the product.
3. In the product details screen, choose [Subscribe](#).

You can subscribe to:

- [Add to Existing Application](#): The list of applications appears. Choose the required application.
- [Create New Application](#): Create an application by entering the name and title. The selected Product is added to the application by default.

4. Choose [Save](#) .

## 1.8.10 Subscribe to a Product [New Design]

You can subscribe to a product and add it to an existing application or create a new application.

### Context

On the homepage, you can find the list of products under various categories. You can also view the various resources that each product has to offer.

#### Note

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Subscribe to a Product \[Classic Design\] \[page 685\]](#).

### Procedure

1. Log on to the API business hub enterprise.
2. You can either look for the product under various categories, or use the search bar to search for the product.
3. In the product details screen, choose *Subscribe*. You can subscribe to:
  - *Add to Existing Application*: The list of applications appears. Choose the required application.
  - *Create New Application*: Create an application by entering the name and title. The selected Product is added to the application by default.
4. Choose *Save*.

## 1.8.11 Create an Application [Classic Design]

Create an Application to consume the required APIs.

#### Caution

Effective June 2024, the classic design of the API business hub enterprise will be deprecated and will no longer be accessible. The new design of the API business hub enterprise will be set as your default design from March 2024. For more information, see [Configure the API business hub enterprise \[New Design\] \[page 675\]](#).

#### Note

This document describes the classic design of the API business hub enterprise. To view the documentation for the new design, see [Create an Application \[New Design\] \[page 693\]](#).

## Prerequisites

- You either have the `AuthGroup.API.ApplicationDeveloper` role or `AuthGroup.API.Admin` role assigned to you. For more information on roles, see .

### Note

The `AuthGroup.API.ApplicationDeveloper` role must not be assigned manually to a user from the SAP BTP Cockpit and this role must not be a part of any user group assignment.

This role is assigned by default to a user who onboards to the API business hub enterprise using the Self-registration process or via Add User flow.

A user must be onboarded to API business hub enterprise only via Self-registration or Add User flow. For more information on registering in API business hub enterprise, see [Register on API business hub enterprise \[page 664\]](#). In the Add User flow, the API business hub enterprise admin adds a user who wants to be onboarded to API business hub enterprise. However, the user who is requesting to be onboarded must ensure that the user details provided to the admin matches the user details obtained from the response of `<developer portal url>/api/1.0/users`.

An application is a discrete representation of the actual developer's application. It provides the developer with an API key to pass-in with every request to the API.

In API Management, similar APIs are bundled together to form products, which are published in the Catalog. An application developer enters necessary details to register to the API business hub enterprise. After successful registration, the Application Developer can explore the required products and APIs to create an application. Once the application has been created successfully, the system generates an Application Key and Application Secret. If APIs in the application you created are protected via Verify API Key policy, then to access those APIs, you must pass the generated Application Key. Whereas, if APIs are protected via OAuth policy, then to access those APIs, you must pass an OAuth token that can be obtained by using the combination of generated Application Key and Application Secret.

## Creating an Application with Application Developer Role

1. Log on to the API business hub enterprise.
2. Navigate to the [My Workspace](#) page.  
If you have created applications earlier, they're displayed under the [Applications](#) section. For a created application, you can view the total number of calls made in the current month.

### Note

By default, the [Cost](#) section displays the cost incurred in the last 6 months and the cost incurred in the current month. However, you can choose a month to view the cost incurred for that month.



You can choose  to obtain the latest metering data.


### Note

You might experience some delay before you see the latest metering data.


Notation used to display the data is as per metric specifications, for example:

- 999 shows as 999 and 1000 shows as 1k
- 999000 shows as 999K and 1000000 shows as 1M
- 1500000 shows as 1.5M and 1000000000 shows as 1G



3. To create an application, choose  in the *Applications* section.
4. In the *Create an Application* dialog, enter a **Title**, a **Description** (optional), and a **Callback URL** (optional) for the application.



5. Choose  to add products to this application.
6. In the *Add Products* dialog, select the products that you want to associate with the application.

#### Note

You can select multiple products published from the same portal but you can't select products published from different portals. For example, you can select products P1 and P2 from API portal A1. But you can't choose products, P3 and P4 from API portal A2 if you've already selected P1 and P2 from A1.

7. Choose *OK*.
8. Choose *Save*.

#### Note

While creating an application, if you've selected the *Take me to this new application now* checkbox, you're directly navigated to the newly created application.

The application you have created appears under the *Applications* section, and also under the *Applications* tab in API portal.

#### Note

If you open any created application, you notice that the system has generated an *API Key* automatically. Use this value to access the API. At any point in time, you can regenerate the API Key using *Regenerate Key* option. When you regenerate the key, both Application key and Secret key are changed. When you trigger API using the old key, then the response is negative. The old API key becomes invalid on regeneration.

## Creating an Application with API business hub enterprise Administrator Role

A API business hub enterprise administrator can perform the following tasks:

- Create an application on behalf of a user (Application Developer) and handover the application key and secret to that user.
- Create new applications in different landscapes(example: production, nonproduction) by maintaining the same application key and secret.



- Create custom attributes at application level and regulate the API call logic
1. Log on to the API business hub enterprise.
  2. Navigate to the *My Workspace* page.

The screenshot shows the 'My Workspace' interface. The 'Applications' table lists the following data:

Application	Developer	Total Calls	Actions
S4 hana app		0	[Trash]
Billing App		6	[Trash]
aap		0	[Trash]
_new_test		3	[Trash]
AA_test_app_new		0	[Trash]
AA_test_app		14	[Trash]
AA_test_app			
multi_new_test		0	[Trash]
wertwestfve		0	[Trash]
wer			
app-001		0	[Trash]

The 'Cost' section shows a bar chart for 'Aggregated Costs in Euros' with a single bar for August reaching 80. The 'Cost Incurred in the Current Month in Euros' table shows:

Application	Cost
S4 hana app	40
S4_hana_prod: 40	
Billing App	40
BllsProd: 40	
<b>Total</b>	<b>80</b>

If you or other application developers have created applications earlier, they're displayed under the *Applications* section. For a created application, you can view the total number of calls made in the current month.


#### Note

By default, the *Cost* section displays the cost incurred in the last 6 months and the cost incurred in the current month. However, you can choose a month to view the cost incurred for that month.

#### Note

For API business hub enterprise administrators, analytics data is unavailable for those applications that they created on behalf of other users or application developers.



You can choose  to obtain the latest metering data.


#### Note

You might experience some delay before you see the latest metering data.

Notion used to display the data is as per metric specifications, for example:

- 999 shows as 999 and 1000 shows as 1k
- 999000 shows as 999K and 1000000 shows as 1M
- 1500000 shows as 1.5M and 1000000000 shows as 1G




3. To create an application, under *Applications* section, choose .
4. In the *Create an Application* dialog, enter a **Title**, a **Description** (optional), and a **Callback URL** (optional) for the application.
5. As an administrator, you have the option to create an application on behalf of a user (Application Developer). To achieve this task, select the *Create this application on behalf of someone else* checkbox and enter the **User ID** of the user on behalf of whom you are creating the application. If you already possess an application key and secret, then select the *Already have Application Key and Secret* checkbox and enter the **Application Key** and **Application Secret**.

#### Note

Application key and secret of an existing org can't be used in a new application in a new org. This implies that you'll not be able to use the same application key and secret in multiple orgs within the same data center and region.



6. Choose  to add products to this application.
7. In the *Add Products* dialog, select the products that you want to associate with the application.

#### Note

You can select multiple products.

8. Choose *OK*.
9. Choose *Save*.

#### Note

While creating an application, if you've selected the *Take me to this new application now* checkbox, you're directly navigated to the newly created application.

The application you created appears under the *Applications* section, and also under the *Applications* tab in API portal.

#### Note

If you open any created application, you notice that the system has generated an *API Key* automatically. Use this value to access the API. At any point in time, you can regenerate the API Key using *Regenerate Key* option. When you regenerate the key, both Application key and Secret key

are changed. When you trigger API using the old key, then the response is negative. The old API key becomes invalid on regeneration.

10. Specify custom attributes.

1. Under *My Workspace*, choose an application for which you want to add custom attributes.

2. In the *Application Info* screen, under *Custom Attributes* section, choose  to add a custom attribute.

3. In the *Add Custom Attribute* dialog, enter a name and a value for your custom attribute and choose *Add*.

#### Note

You can create a maximum of 18 custom attributes per application. You cannot modify the name of a created custom attribute. However, you can modify its value whenever required. You can delete a custom attribute if it is no longer needed.

For more information on the usage of custom attributes in an application, see [Example: Accessing the Custom Attributes of an Application \[page 691\]](#).

## 1.8.11.1 Example: Accessing the Custom Attributes of an Application

Let's say as a Developer Portal Administrator, you would want to restrict the number of calls to an application based on Application Key. To achieve this result, you create two applications `Application_1` and `Application_2`.

`Application_1` contains two products namely `Prod_1` and `Prod_2`.

`Application_2` contains two products namely `Prod_3` and `Prod_4`.

`Prod_1` and `Prod_2` contain two common APIs namely `API_1` and `API_2`.

For `Application_1`, add the following custom attributes and its corresponding values:

- `app_time_unit` = minute
- `app_quota_interval` = 1
- `app_quota_count` = 9

For `Application_2`, add the following custom attributes and its corresponding values:

- `app_time_unit` = minute
- `app_quota_interval` = 1
- `app_quota_count` = 5

To leverage these custom attributes in your API proxy execution, you must:

- add a verify API Key policy to the APIs that are part of your application.
- add a Quota policy to APIs that are part of your application.

For API\_1 and API\_2, add the following sample policy payloads:

Sample payload for Verify API Key policy:

#### Sample Code

```
<!--Specify in the APIKey element where to look for the variable containing
the api key-->
<VerifyAPIKey async='true' continueOnError='false' enabled='true'
xmlns='http://www.sap.com/apimgmt'>
 <APIKey ref='request.queryparam.apikey' />
</VerifyAPIKey>
```

Sample payload for Quota policy:

#### Sample Code

```
<!-- can be used to configure the number of request messages that an app is
allowed to submit to an API over a course of unit time -->
<Quota async="false" continueOnError="false" enabled="true" type="calendar"
xmlns="http://www.sap.com/apimgmt">
 <Identifier ref="verifyapikey.Verify-api-key.developer.id"/>
 <!-- specifies the number of requests allowed for the API Proxy -->
 <Allow countRef="verifyapikey.Verify-api-key.app_quota_count"/>
 <!-- the interval of time for which the quota should be applied -->
 <Interval ref="verifyapikey.Verify-api-key.app_quota_interval"/>

 <!-- used to specify if a central counter should be maintained and
continuously synchronized across all message processors -->
 <Distributed>true</Distributed>
 <!-- Use to specify the date and time when the quota counter will begin
counting,
 regardless of whether any requests have been received from any apps
-->
 <StartTime>2015-2-11 12:00:00</StartTime>
 <!-- if set to true, the distributed quota counter is updated
synchronously. This means that
 the update to the counter will be made at the same time the API call
is quota-checked -->
 <Synchronous>true</Synchronous>
 <!-- Use to specify the unit of time applicable to the quota. Can be
second, minute, hour, day, or month -->
 <TimeUnit ref="verifyapikey.Verify-api-key.app_time_unit"/>
</Quota>
```

#### Note

The attribute names `app_quota_interval`, `app_quota_count`, and `app_time_unit` must be the same attributes that you have added while creating the application.

To verify if the custom attributes are used in runtime, make an API call with `<appKey_1>` passed as a query parameter. For example, `https://<API_proxy_URL>?apikey=<appKey_1>`.

Call the same URL repeatedly and after 9 successive calls, your API proxy must return a Quota violation message.

Similarly, make an API call with `<appKey_2>` passed as a query parameter. For example, `https://<API_proxy_URL>?apikey=<appKey_2>`.

Call the same URL repeatedly and after 6 successive calls, your API proxy must return a Quota violation message.

## 1.8.12 View Applications, Costs, and Analyze Reports [New Design]

From the [My Workspace](#) page you can view your applications, costs, and analyze reports.

Applications created by you or any other application developers are displayed under the [Applications](#) section. For a created application, you can view the total number of calls made in the current month. From this page, you can create a new application or create an application on behalf of a user, mostly application developers.

By default, the [Cost](#) section displays the cost incurred in the last 6 months and the cost incurred in the current month. However, you can choose a month to view the cost incurred for that month.

You can analyze the performance of all your applications and get an overview of the application usage from the [Performance Analytics](#) section. In this section the runtime data is gathered, analyzed, and displayed as charts, headers, and key performance indicators (KPIs). For more information, see [Analyze Applications \[page 698\]](#).

You can also navigate to the [Error Analytics](#) tab to view the error-related charts and KPIs for all the applications for the selected time period.

## 1.8.13 Create an Application [New Design]

Create an Application to consume the required APIs.

### Note

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Create an Application \[Classic Design\] \[page 686\]](#).

An application is a discrete representation of the actual developer's application. It provides the developer with an API key to pass-in with every request to the API.

In API Management, similar APIs are bundled together to form products, which are published in the catalog. An application developer enters necessary details to register to the API business hub enterprise. After successful registration, the application developer can explore the required products and APIs to create an application. Once the application has been created successfully, the system generates an application key and application secret. If APIs in the application you created are protected via **Verify API Key** policy, then to access those APIs, you must pass the generated application key. Whereas, if APIs are protected via **OAuth** policy, then to access those APIs, you must pass an OAuth token that can be obtained by using the combination of generated application key and application secret.

A user must be onboarded to API business hub enterprise only via Self-registration or [Add User](#) flow. For more information on registering in API business hub enterprise, see [Register on API business hub enterprise \[page 664\]](#). In the [Add User](#) flow, the API business hub enterprise admin adds a user who wants to be onboarded to API business hub enterprise. However, the user who is requesting to be onboarded must ensure that the user details provided to the admin matches the user details obtained from the response of <developer portal url>/api/1.0/users.

## 1.8.13.1 Creating an Application with Application Developer Role [New Design]

As an application developer you can create an application, and view the existing application details, and its associated products, custom attributes and analytics data.

### Prerequisites

- You have the [AuthGroup.API.ApplicationDeveloper](#) role assigned to you. For more information on roles, see .

#### Note

The [AuthGroup.API.ApplicationDeveloper](#) role must not be assigned manually to a user from the SAP BTP Cockpit. Also, this role must not be a part of any user group assignment.

The [AuthGroup.API.ApplicationDeveloper](#) role is assigned by default to a user who onboards to the API business hub enterprise using the self-registration process or via [Add User](#) flow. For more information on registering in API business hub enterprise, see [Register on API business hub enterprise \[page 664\]](#).

In the Add User flow, the API business hub enterprise admin adds a user who wants to be onboarded to API business hub enterprise. However, the user who is requesting to be onboarded must ensure that the user details provided to the admin matches the user details obtained from the response of `<developer portal url>/api/1.0/users`.

### Context

You are about to create an application and add products to your application. You can also select an existing application and view its details under the [Application Details](#) tab. Navigate to the [Products](#) tab to view the products and the rate plan associated with this application. You can add custom attributes to your applications, and manage them from the [Custom Attributes](#) tab. For more information, see [Add Custom Attributes to an Application \[page 612\]](#). Navigate to the [Analytics](#) tab to analyze application usage, performance, and error count. For more information, see [Analyze Applications \[page 698\]](#).

### Procedure

- Log on to the API business hub enterprise and navigate to [My Workspace](#).  
The applications you created earlier, are displayed under the [Applications](#) tab. For an existing application, you can view the total number of calls made in the current month on the [Applications](#) page.
- To create an application, choose [Create New Application](#) in the [Applications](#) section.
- In the [Create an Application](#) dialog, enter a [Title](#), a [Description](#) (optional), and a [Callback URL](#) (optional) for the application.

You can also choose the options [Create this application on behalf of someone else](#) or [Already have Application Key & Secret](#).

#### Note

While creating the application, if you've selected the [Take me to this new application now](#) checkbox, you're directly navigated to the newly created application.

4. To add products to this application, choose [Add Products](#).
5. In the [Add Products](#) dialog, select the products that you want to associate with the application.

#### Note

You can select multiple products published from the same portal but you can't select products published from different portals. For example, you can select products P1 and P2 from API portal A1. But you can't choose products, P3 and P4 from API portal A2 if you've already selected P1 and P2 from A1.

6. Choose [Create](#).

You can find the details of the application you just created under the [Application Details](#) tab.

#### Note

The system generates an [Application Key](#) automatically when an application is created. You should use this application key value to access the API. At any point in time, you can regenerate the application key using [Regenerate Key](#) option. When you regenerate the key, both the application key and the secret key are changed. When you trigger API using the old key, then the response is negative. The old application key becomes invalid on regeneration.

7. To add a custom attribute, choose [Custom Attributes > Add Custom Attributes](#).
8. In the [Add Custom Attribute](#) dialog, enter a name and a value for your custom attribute and choose [Add](#)

#### Note

You can create a maximum of 18 custom attributes per application. You cannot modify the name of a created custom attribute. However, you can modify its value whenever required. You can delete a custom attribute if it is no longer needed.

For more information on the usage of custom attributes in an application, see [Example: Accessing the Custom Attributes of an Application \[page 691\]](#).

## 1.8.13.2 Creating an Application with API business hub enterprise Administrator Role [New Design]

With the API business hub enterprise administrator role you can create an application on behalf of a user (application developer), and view the existing application details, and its associated products, custom attributes, and analytics data.

### Prerequisites

You should have the [AuthGroup.API.Admin](#) role assigned to you. For more information on roles, see .

### Context

An API business hub enterprise administrator can perform the following tasks:

- Create an application on behalf of a user (Application Developer) and handover the application key and secret to that user.
- Create new applications in different landscapes (example: production, nonproduction) by maintaining the same application key and secret.
- Create custom attributes at application level and regulate the API call logic.

### Procedure

1. Log on to the API business hub enterprise and navigate to [My Workspace](#).

If you or other application developers have created applications earlier, they're displayed under the Applications section. For a created application, you can view the total number of calls made in the current month.

#### Note

For API business hub enterprise administrators, analytics data is unavailable for those applications that they created on behalf of other users or application developers.

2. To create an application, choose [Create New Application](#) in the [Applications](#) section.
3. In the [Create an Application](#) dialog, enter a [Title](#), a [Description](#) (optional), and a [Callback URL](#) (optional) for the application.

As an administrator, you have the option to create an application on behalf of a user (application developer). To achieve this task, select the [Create this application on behalf of someone else](#) checkbox, and enter the [User ID](#) of the user on behalf of whom you are creating the application. If you already possess an application key and secret, then select the [Already have Application Key and Secret](#) checkbox and enter the [Application Key](#) and [Application Secret](#).



### Note

Application key and secret of an existing org can't be used in a new application in a new org. This implies that you'll not be able to use the same application key and secret in multiple orgs within the same data center and region.

### Note

While creating the application, if you've selected the *Take me to this new application now* checkbox, you're directly navigated to the newly created application.

- To add products to this application, choose *Add Products*.
- In the *Add Products* dialog, select the products that you want to associate with the application.

### Note

You can select multiple products published from the same portal but you can't select products published from different portals. For example, you can select products P1 and P2 from API portal A1. But you can't choose products, P3 and P4 from API portal A2 if you've already selected P1 and P2 from A1.

- Choose *Create*.  
You can find the details of the application you just created under the *Application Details* tab.
- To add a custom attribute, choose **► Custom Attributes ► Add Custom Attributes ►**.
- In the *Add Custom Attribute* dialog, enter a name and a value for your custom attribute and choose *Add*

### Note

You can create a maximum of 18 custom attributes per application. You cannot modify the name of a created custom attribute. However, you can modify its value whenever required. You can delete a custom attribute if it is no longer needed.

For more information on the usage of custom attributes in an application, see [Example: Accessing the Custom Attributes of an Application \[page 691\]](#).

## 1.8.14 Consume Applications

Once you create an Application, you can then consume the APIs based on your business requirements.

On subscribing to an application, the application developer receives an API key that the application must pass on every request to the API. API keys provide a simple mechanism for authenticating applications.

API Management generates API keys for applications, and enables you to add API key-based authentication to your APIs using policies. However, enforcement of the key is performed at the API proxy level, not by the API product itself. Therefore, you must ensure that all API proxies, and the corresponding resources defined by those API proxies, implement some form of key validation.

Before you use the API keys, ensure you are aware of the policies that support API keys and their functionality. There are two popular ways how APIKeys are provisioned. They are provided either as part of a Simple APIKey verification or as part of OAuth verification.

## VerifyAPIKey Validation

If you define an API proxy to perform key validation by using the VerifyAPIKey policy, provide the API Key details to gain access to the applications.

## OAuth 2.0 Validation

API Management supports standard OAuth flow. Currently SAP supports only Client credentials grant\_type in OAuth.

Before you start, make a note of the Application key and secret for the required application.

A request is made using the following:

- URL: <URL of OAuth token>
- Method: POST
- Custom Header: Authorization value: Basic <Application key>:<Application secret>(base64 encoded)
- Payload: grant\_type=client\_credentials

This call returns a json payload with the OAuth validation response. On successful validation, it contains the access token. Note down the access token.

As default, the expiry time is configured to 3600 secs (1 hr). You can also configure the expiration time and the details that have to be displayed as part of the response.

You can now use this access token to fetch OAuth enabled services while making the actual business API call:

- URL: http[s]://<host>:<port>/<service\_path>
- Custom Header: Authorization value: Bearer <access\_token>

If the access token is valid, a valid service response is returned.

## 1.8.15 Analyze Applications

Use analytics capabilities to analyze application usage, performance, and error count.

API Management provides comprehensive analytics capabilities to understand application consumption. The runtime data is gathered, analyzed, and displayed as charts, headers, and key performance indicators (KPIs).

As an application developer, navigate to [My Workspace](#) to view the analytics information. By default, the analytics section displays the data for all the applications subscribed by you. All charts are displayed based on the Application Developer context.

The analytics information can be viewed as [Performance Analytics](#) and [Error Analytics](#).

- **Performance Analytics:** Displays the performance-related charts and KPIs for the selected time period. Following table describes the charts used to analyze the performance of all applications:

## Performance Analytics

Chart Name	Description
Traffic Across all APIs	This chart displays total API calls made across all applications.
Slowest APIs	This chart displays the slowest APIs based on the API response time.
Top APIs	This chart displays most frequently used APIs.
Top Products	This chart displays most frequently used products based on the number of calls made to the APIs associated with the product.
Top Applications	This chart displays most frequently used applications based on the number of calls made to the APIs associated with the application.

- **Error Analytics:** Displays the error-related charts and KPIs for the selected time period. Following table describes the charts used to view error analytics of all the applications:

## Error Analytics

Chart Name	Description
Total Errors	This chart displays total errors.
Error Prone APIs	This chart displays number of errors per API.
Error Prone Applications	This chart displays number of errors per API associated with the application.

To view analytics for a specific application, navigate to the application details screen by selecting the required application. However, in the application details screen the analytics information is available only for the following KPIs:

- Traffic Across all APIs
- Slowest APIs
- Error Prone APIs

## 1.8.16 Consume API Proxies Using SAP Business Application Studio

The service center in SAP Business Application Studio provides a central entry point to explore products and services from the API business hub enterprise.

You can use this service center to develop your applications based on the OData Services available as a part of products published in API business hub enterprise. For more information, see [API Business Hub Enterprise Service Provider](#).

## 1.8.17 Test Runtime Behavior of APIs [New Design]

Use the API Test Environment to test the runtime behavior of APIs.

The [Test Environment](#) enables you to test your APIs. Testing an API is essential to understand the runtime behavior of the APIs. It allows you to explore the resources associated with an API and execute the operations. It also allows you to test OData and REST-based services.

## ⓘ Note

This document describes the new design of the API business hub enterprise. To view the documentation for the classic design, see [Test API Proxies \[page 648\]](#).

## Pre-requisite

The *Test Environment* tab will be visible to you only if you have the *AuthGroup.API.ApplicationDeveloper* role

## Procedure

1. Log on to the API business hub enterprise.
2. Navigate to the *Test Environment*.
3. A list of APIs appears on the left.
4. Select the required API.  
The URL for the selected API is populated automatically in the *API Test Console*. For the selected API, the URLs of the supported resources appear in the dropdown list. One resource is selected by default.
5. If you want to choose a different collection, use the dropdown list to select the required collection
6. If you have the URL of the service that contains the API, enter the service URL.
7. Choose *Authentication* to select the required type of authentication. You can choose from the following options:
  1. *None*: No authentication required.
  2. *Basic Authentication*: Provide a user name and password.
8. Enable the required method:
  - *GET*: Reads an entity
  - *POST*: Creates an entity
  - *PUT*: Updates an entity
  - *DELETE*: Deletes an entity

## ⓘ Note

You can enable only the methods supported by the service.

9. Enter the *Request Body* for PUT and POST methods.
10. Choose *Header* to add a header.

## ⓘ Note

If you want to add multiple headers, choose *Add Request Headers*.

11. Choose *Url Params* to enter the query parameter and value.

### Note

If you want to add multiple query parameters, choose the button [Add URL Params](#). Test Console supports passing of custom headers such as X-sap-apimgt-proxy-host:-proxy-trai and X-sap-apimgt-proxy-port:- 8080

12. Choose [Send](#).

The response appears in the tabs:

- [Body](#): View the formatted response.
- [Body \(Raw\)](#): View the unformatted response.
- [Headers](#): View the headers.
- [Cookies](#): View the cookies.

13. If you want to use the response body as an input request, choose [Use as Request](#) on the [Body \(Raw\)](#) tab.

14. To view the transactions based on the testing activity that you did, choose [Launch API Viewer](#). For more information on tracing API proxy, see [Debug an API Proxy \[page 650\]](#)

## 1.9 Analyze APIs

Use the capabilities of API Analytics to analyze API usage and performance.

provides comprehensive analytics capabilities to understand the various patterns of API consumption and performance. The API Analytics server uses the runtime data of the APIs to analyze the information. The runtime data is gathered, analyzed, and displayed as charts, headers, and key performance indicators (KPIs).

Use the analytics dashboard to view the aggregated results. The detailed analytics view helps to manage APIs, attract the right application developers, troubleshoot problems, and ultimately, make better business decisions.

There are two variants of analytics dashboard available for analyzing API reports:

- **API Analytics**

The analytics dashboard provides a comprehensive view of analytical charts and KPIs relevant to the performance of the APIs. The dashboard also displays the error-related charts and KPIs such as total number of API errors, total number of system errors, and policy errors. For more information, see [API Analytics \[page 702\]](#).

- **Advanced API Analytics**

Advanced API Analytics brings to you the all new analytics dashboard, providing powerful tools and in-depth reports for analyzing your API usage and performance. The reports are categorized across report pages, with each report page providing information about key API metrics, which are relevant for both business users and API developers. For more information, see .

### Note

We're calling this new flavor of analytics, Advanced API Analytics. However, for you, as an end-user there will be no direct reference to this title on the user interface. We're sure that the various new features will enable you to make best use of this exciting new analytics dashboard.

## Related Information

[Analytics Dashboard \[page 702\]](#)

[Find Your Way around Advanced API Analytics Dashboard \[page 706\]](#)

### 1.9.1 API Analytics

API Analytics provides sample analytical charts and key performance indicators (KPIs). These charts and KPIs are preconfigured in the dashboard.

Examples of standard analytical charts include:

- What is the API traffic trending over time?
- Which five APIs have the slowest response time?
- What is the overall API error count?

Examples of standard KPIs include:

- Total API hits
- API response time
- Total number of policy errors

## Related Information

[Analytics Dashboard \[page 702\]](#)

[Working with the Analytics Dashboard \[page 703\]](#)

### 1.9.1.1 Analytics Dashboard

The analytics dashboard has some common features such as the views you can choose, the time range for which you want to display data, resize charts, and so on.

## Features of the Analytics Dashboard

The following lists common features on the dashboard:

1. **Views:** The dashboard provides three views:
  - **Performance View:** Displays the performance-related charts and KPIs, such as API traffic for a specific period of time.
  - **Error View:** Displays error-related charts and KPIs, such as total number of API errors.

- **Custom View:** Displays custom charts that you created by selecting measures, dimension, and chart type.
2. **Time interval:** Displays data only for the selected period of time. For example, time interval of months, weeks, and so on.
  3. **Resize charts:** Resize charts from small, medium to large.

#### 📌 Note

The values on the chart for a particular dimension are shown only up to a certain threshold. Any values beyond this threshold are grouped together and labelled as **Others**. By default, the threshold value for the charts is set to **25**. If you wish to modify the threshold value for the charts, please create a support ticket. To create a support ticket, see [Request to Modify Threshold Value for Charts \[page 716\]](#).

## Related Information

[Working with the Analytics Dashboard \[page 703\]](#)

### 1.9.1.2 Working with the Analytics Dashboard


The analytics dashboard provides a comprehensive view of API performance and errors in the form of charts and KPIs.

## Context

You can use the analytics dashboard to:

- View the performance of all APIs
- View the errors of all APIs
- Filter data based on a selected period

## Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose .

The *Analytics* dashboard appears.

3. Use the dropdown list on the top left to select one of the following views:
  - **Performance View:** Displays performance-related charts and KPIs. For example, you can display a chart showing API traffic, or view performance-related KPIs such as total number of API hits, average response time for APIs, and so on.

- **Error View:** Displays error-related charts and KPIs. For example, you can display a chart showing the top 5 API policy-related errors, or view error-related KPIs such as total error count, total number of system errors, total number of API policy errors, and so on.
  - **Custom View:** Displays custom charts that you've created by selecting measures, dimensions, and the chart type from the pane on the left.
4. You can perform the following actions for every chart on the dashboard:
    - **Details:** Navigate to the details page of any chart by choosing *Details* in the top-right corner of the chart. On the details page, you can switch between different chart types and apply filters.
    - **Resize:** Change the size of the chart (small, medium, or large).
  5. If you want to analyze data for a selected period, choose the required time period at the top of the dashboard. You can view data for the following time periods:
    - Last day
    - Last 7 days
    - Last 30 days
    - Last 6 months
    - Custom range (date interval and frequency) according to your requirements

#### Note

The data retention period in the analytics dashboard is 6 months. That is, the analytics dashboard stores and retains data only for a period of 6 months. After the retention period, the data is purged.

6. If you want to customize an analysis using specific measures and dimensions, see [Customizing an Analysis \[page 704\]](#)

## 1.9.1.2.1 Customizing an Analysis

The analytics dashboard allows you to customize an analysis using specific dimensions and measures.

### Context

You can create custom charts using selected dimensions and measures to analyze the performance of APIs.

### Procedure

1. Select *Analyze Data* in the bottom-right corner of the screen.
2. Enter a chart title of your choice in the *Title* field.
3. Add the required measures and dimensions in the left pane.
4. The selected measures appear in the *Selected Measures* dropdown list. You can select the required measure from this list and plot the chart.



### Note

You can add multiple measures with various dimensions for a single chart. If you add more than two measures, a table with measure details is displayed below the chart.

5. To drill down on a particular dimension, click the corresponding bar on the plotted chart. If you apply a filter to a chart to drill down to the details, you can navigate back to any previous parameter by using the breadcrumb option.
6. If you want to analyze using a particular value of measure or dimension in the plotted chart, choose the Filter icon at the top of the chart and set the required values in the Filter popup. You can enter values for measures manually using the Equals to, Greater than, or Less than options. For static dimensions (default), you can choose multiple values from the dropdown list. For custom dimensions (created by you), you can enter multiple values separated by commas.
7. Choose *OK* and then save the chart.
8. The chart appears in the *Custom View*.

### Note

As you analyze your data, you may see an entity's value of (n.a or not set) displayed, including the parentheses, for your API Proxies, Product, Developer, and Developer Apps dimensions. This could be due to one of the following reasons:

- Not all of your API proxies and developer applications are using products.
- Some of your traffic is generated by unregistered developers and applications. This traffic may originate from an internal-use or public API.

9. To remove a custom chart from custom view, select *Unpin from Custom View* option from the *Chart Settings* dropdown.
10. To add a custom chart, select *Change Analytic View* option from the *Chart Settings* dropdown.
11. To delete a custom chart, select *Delete* option from the *Chart Settings* dropdown.
12. To edit a custom chart, choose *Edit Chart* icon.

### Note

While editing the charts, there's a possibility that the changes that you make to the order of the measures might not get preserved. In such cases, you can follow the steps given below to modify the measures in your charts:

1. Remove all the measures in the chart except for the one you want to view.
2. Save the chart.
3. Edit the chart again to add the next measure and save the changes. Repeat this step to add the other measures to the chart.

## 1.9.2 Advanced API Analytics

Advanced API Analytics brings to you the all new analytics dashboard, providing handy and powerful analytical reporting tools to track your API performance and usage.

### Note

We are calling this new flavor of analytics, Advanced API Analytics. However, for you, as an end-user there will be no direct reference to this title on the user interface. We are sure that the various new features will surely enable you to make best use of this exciting new analytics dashboard.

Most of the reports on the analytics dashboard are a graphical representation of data, derived using visually appealing charts. You can choose between different chart types to visualize data as per your needs. The chart-oriented representation enables you to quickly glance through and analyze important API metrics, thus helping you in making better business decisions. The analytical data is spread across various report pages namely Overview, Health, and Usage, with each page providing information about key API metrics.

- **Overview**

The Overview page provides a concise report about important and key API metrics. In the Overview page, both business users and API developers can quickly analyze reports and view API trends for the last seven days. Business users can obtain information about most popular APIs, total number of API calls, and key application developers. Developers can obtain information about non-performing APIs and the factors affecting these APIs such as response time and latency.

- **Health**

The Health page provides reports about key metrics related to the performance of your APIs. In the Health page, API developers can quickly monitor the API metrics that affect the performance of APIs and view API error trends for the last seven days. The key monitoring metrics include information about average API response time and the common types of API errors. API developers can also obtain information about recent error responses for an API and the total number of erroneous calls.

- **Usage**

The Usage page provides reports about key metrics related to user-engagement. In the Usage page, business users can analyze API metrics that indicate the overall traction or acceptance of their API program and view API trends for the last three months. The key user-engagement metrics include information about the sources or medium from where you are acquiring users and traffic to your APIs. Business users can also obtain information about new developers who are onboarded to their API program, and a list of recently created applications.

### Related Information

[Find Your Way around Advanced API Analytics Dashboard \[page 706\]](#)


### 1.9.2.1 Find Your Way around Advanced API Analytics Dashboard

Familiarize yourself with the main features and controls of the Advanced API Analytics dashboard.

## Help and Notifications Menu

Every page in Analytics dashboard gives you access to notifications, help documentation, and lets you manage your account. This menu is available at the top-right corner of the analytics dashboard.

Click  to know about the latest news and updates for .

Click  to view the version of , access the online help documentation, and logout of .

## Report Pages

In the Analytics dashboard, you access all your reports in report pages. The reports are categorized into report pages namely Overview, Health, and Usage. These report pages provide information about key metrics related to your API usage and performance.

You can find a time zone switcher to the right side of the report pages. The time zone switcher allows you to view analytics data based on different time zones. The default time zone shown is UTC. The time zone switcher is available across all the report pages including custom report pages.

### Note

The values on the chart for a particular dimension are shown only up to a certain threshold. Any values beyond this threshold are grouped together and labelled as **Others**. By default, the threshold value for the charts is set to **25**. If you wish to modify the threshold value for the charts, please create a support ticket. To create a support ticket, see [Request to Modify Threshold Value for Charts \[page 716\]](#).

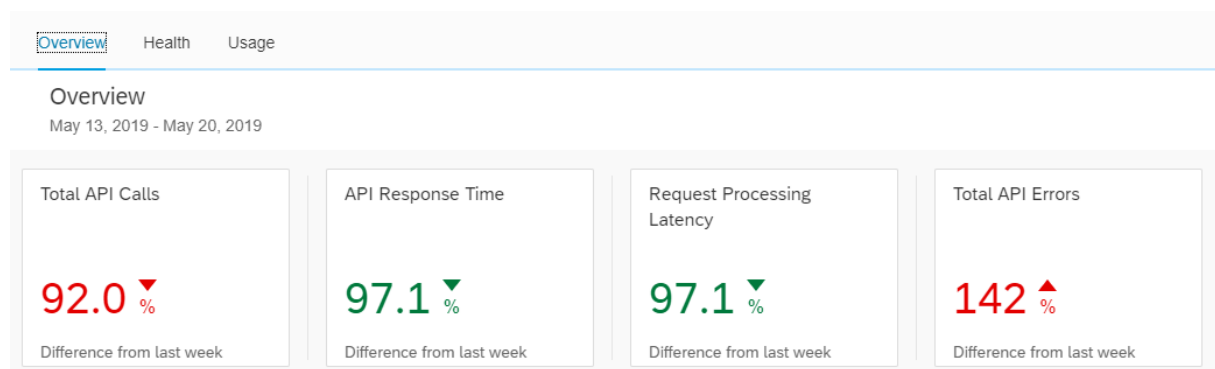
### Note

The data retention period for all report types available in the analytics dashboard is 6 months. That is, the analytics dashboard stores and retains data only for a period of 6 months. After the retention period, the data is purged.

## Overview

The Overview page provides a summarized report about your most important and key API metrics. By default, the Overview page provides report data for the last seven days.

At the top of the Overview report page, the following key API metrics are represented in a tile format:



- **Total API Calls:** Total number of API requests made over a specified period.
- **API Response Time:** Total time in milliseconds to respond to an API request. This round-trip time includes the API proxy overhead and the target server time.
- **Request Processing Latency:** Total time in milliseconds from the time when a request reaches the selected API proxy to the time when the proxy sends the request to the target server.
- **Total API Errors:** Total number of errors that occur on the API proxy plus the errors that occurs on the target server from all the API requests and responses over a specified period.
- **Target System Errors:** Total number of errors that occur on the target server over a specified period.
- **Target Response Time:** Total time in milliseconds for the target server to respond to an API request.

Each tile shows weekly percentage difference in data for a key API metric. A green-arrowed percentage difference indicates a healthy API metric, whereas, a red-arrowed percentage difference indicates the API metric needs improvement.

Hovering over each tile shows data for the current week, last week, and the difference between current week and last week.

Just below the key API metrics tiles, you can filter and check for the number of calls for each of your APIs using the **API Calls** dropdown menu. By default, it shows the total number of success and failure calls for all the APIs, combined.

The rest of the Overview page provides a graphical view of other key API metrics, which include the following:

- Top APIs of the Week
- Top Products of the Week
- Top Applications of the Week
- Top API Providers of the Week
- Top Developers of the Week
- Top APIs
- Top Applications
- Top Products
- Top API Providers
- Top Developers
- Top Response Codes
- API Errors
- API Response Time
- Developer Engagement
- Top Backend Errors
- Top Proxy Errors
- Slowest APIs

### 📌 Note

In any of the key API metrics, if you observe an entity in the chart marked as `Unidentified`, it indicates the following:

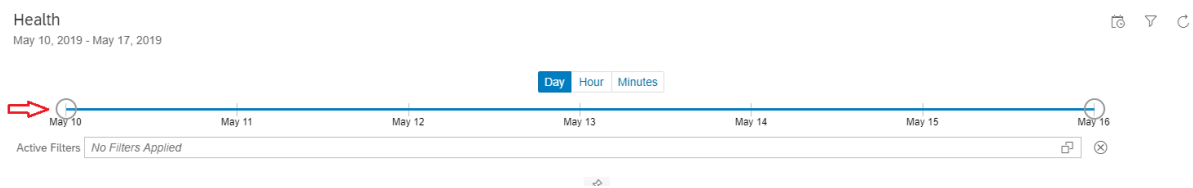
- `Unidentified Application`: An application could not be identified for these API calls as the API key could not be verified.
- `Unidentified Product`: A product could not be identified for these API calls as the API key could not be verified.

- **Unidentified Developer:** A developer could not be identified for these API calls as the API key could not be verified.

## Health

The Health page provides reports about key metrics related to the performance of your APIs. By default, it shows data for the last seven days.

At the top of the Health page, there is a date-range selector. This date-range selector lets you set the time period for which you want to analyze the reports. To set a new time period, click and drag the bubble-like endpoints on the date-range selector.




Above the date-range selector, select **Day**, **Hour**, or **Minutes** tabs to see daily, hourly, or 30-minute aggregate data.

The **Day** option displays seven touch points, one for each day of the week.

The **Hour** option displays 24 touch points, one for each hour of the day.

The **Minutes** option displays 48 touch points, one for every 30 minutes of the day.

At the top right corner of the Health page, click  to view advanced filter menu and options. The filter menu and options appear below the date-range selector and you can filter your reports by API, Applications, Products, or Developers. Once you apply the filter options, the applied filters are displayed under **Active Filters**.

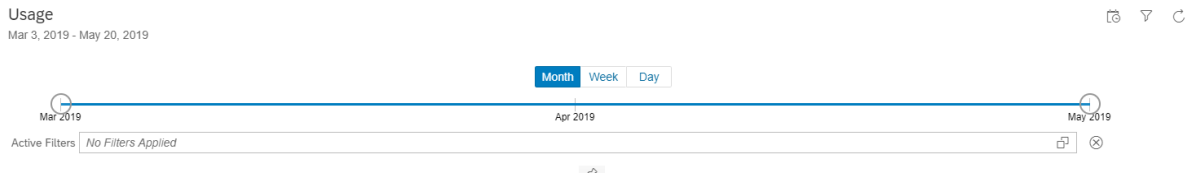
The rest of the Health page displays a graphical view of key API metrics, which include the following:

- API Calls
- Response Code Count
- Cache Responses
- Backend Error Call Count
- Backend Response Time
- API Error Calls
- API Response Time
- Error Count per Response Code
- Average Response Time

## Usage

The Usage report page provides reports on key metrics related to user-engagement. You can obtain information about the sources or medium from where you are acquiring users and traffic to your APIs, your most popular developers applications, and request verbs. By default, it displays data for the last two months and present month until the current date.

At the top of the Usage report page, there is a date-range selector. This date-range selector lets you set the time period for which you want to analyze the reports. To set a new time period, click and drag the bubble-like endpoints available on the date-range selector.




Above the date-range selector, you can select **Month**, **Week**, or **Day** tabs to see data by month, week, or day.

The **Month** option displays three touch points, one for each of the last three months inclusive the current month.

The **Week** option displays one touch point for each week of the last three months inclusive the current month. A week starts on a Sunday and ends on a Saturday.

The **Day** option displays one touch point for each day. The number of touch points displayed here varies depending upon the time range you have selected under Month or Week tabs.

At the top right corner of the Usage page, click  to view advanced filter menu and options. The filter menu and options appear below the date-range selector and you can filter your reports by API, Applications, Products, or Developers. Once you apply the filter options, the applied filters are displayed under [Active Filters](#).

The rest of the Usage page displays a graphical view of key API metrics, which include the following:

- API Calls
- Developer Engagement Status
- New Developers
- New Applications
- Top Browsers
- Top Agents
- Top Operating Systems
- Top Device Types
- Browser Call Count
- Agent Call Count
- Operating Systems Call Count
- Device Types Call Count
- Request Verb Call Count

### Note

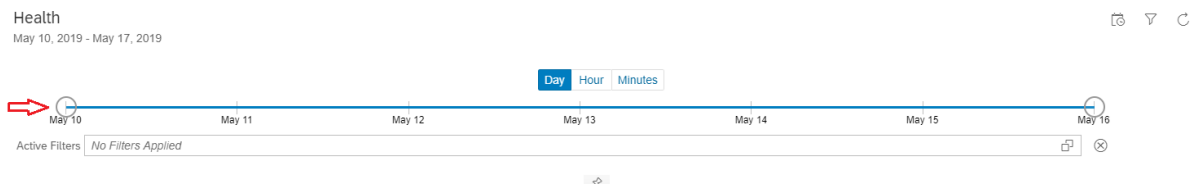
In any of the key API metrics, if you observe an entity in the chart marked as `unidentified`, it indicates the following:

- `Unidentified Application`: An application could not be identified for these API calls as the API key could not be verified.
- `Unidentified Product`: A product could not be identified for these API calls as the API key could not be verified.
- `Unidentified Developer`: A developer could not be identified for these API calls as the API key could not be verified.
- `Unidentified Platform`: A platform could not be identified for these API calls as the API key could not be verified.

- **Unidentified Agent:** An agent could not be identified for these API calls as the API key could not be verified.
- **Unidentified Operating System:** An operating system could not be identified for these API calls as the API key could not be verified.
- **Unidentified Device Type:** A device type could not be identified for these API calls as the API key could not be verified.


## Date-Range Selector


In the **Health** and **Usage** report pages, there is a date-range selector. This date-range selector lets you set the time period for which you want to analyze the reports. To set a new time period, click and drag the bubble-like endpoints available on the date-range selector.



At the top-right corner of the date-range selector, there is a small action bar with options to hide the date-range selector and refresh the reports.

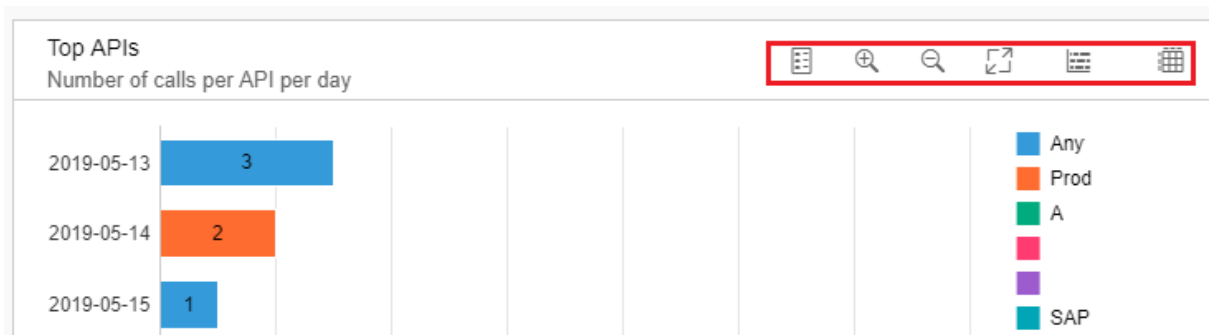
Click  to hide or unhide the date-range selector.

Click  to refresh the reports with latest data from API calls.

While viewing your reports on the Health and Usage page, you can choose to keep the data-range selector always visible. You can do so by clicking on the  icon available below the date-range selector.

## Action Bar


The Action Bar appears at the top of each report that contains graphical data. The controls on the action bar allow you to act on the graphical data. Using these controls, you can switch between graphical view and tabular view, and switch between different chart types.



Click  to hide and unhide legends for a graph.

Click   to view enlarged and diminished image of a graph.

Click  to switch between full screen view and default screen view.

Click  to select a different chart type. You can select between pie charts, line charts, bar charts, donut charts, or heatmaps. Note that not all chart types might be supported for a specific report.

Click  to switch between a tabular view and graphical view of a report.

## Related Information

[Create and Work with Custom Reports \[page 712\]](#)

[Create Custom Dimensions and Measures \[page 714\]](#)

## 1.9.2.2 Create and Work with Custom Reports

Create your own custom reports in Advanced API Analytics dashboard.

### Context

You can create customized charts for API metrics that are critical to your business. Using the Custom view feature, you can group all these API metrics and view them in a single window.

### Procedure

1. In the analytics dashboard, choose *Custom View* from the *+Add* dropdown menu.
2. In the *Create Custom View* dialog, enter a name for your new custom report and choose *OK*.
3. In the *Create Chart* window, enter a title and a description for the chart that you want to create.
4. Under *Dimensions*, choose the API metric that you want to measure from the dropdown menu.
5. Under *Measures*, choose how you want to measure the selected API metric.

For example, you want to create a chart to plot the number of API calls received through a device type. In this case, you can name the chart as Number of calls per device, and choose Device type under Dimensions, and choose Calls under Metrics.

#### Adding multiple dimensions and measures

You can add multiple measures with various dimensions for a single chart. For example, you can plot the total number of calls and errors occurring on a particular device type. To do so, you can select device type under Dimension and add two entries under Measures, one for tracking the number of calls and the other for tracking the number of errors.



To view the chart for a particular measure, select the measure from *Selected Measures* dropdown menu available at the top of the chart. In addition, the dimension and measures, and its details appear as a table below the chart.


You can also add multiple dimensions. For example, if you want to plot the number of calls from a particular device type and from an operating system, then you can add two entries under Dimensions, one for the device type and the other for the Operating system type.

To drill down on a particular dimension, click the corresponding bar on the plotted chart. If you apply a filter to a chart to drill down to the details, you can navigate back to any previous parameter by using the breadcrumb option.

If you want to analyze using a particular value of measure or dimension in the plotted chart, choose the Filter icon at the top of the chart and set the required values in the Filter popup. You can enter values for measures manually using the Equals to and Not Equals to options. For static dimensions (default), you can choose multiple values from the dropdown list.

6. Choose *OK* and then save the chart by choosing *Save*.

The newly created custom view appears as a new report page next to the default report pages. To view all your custom charts and reports, choose the created custom view report page. You can create a maximum of three custom views.

To edit the name of a custom view, select the required custom view and click on the  icon displayed next to the custom view name.

7. To add new charts to your custom view, choose the required custom view and click *Create*.
8. In the *Create Chart* window, enter a name and a description for the chart in the *Title* and *Description* fields, and under *Dimensions and Measures*, choose the API metric that you want to track and measure.

At the top of your custom report page, there is a date-range selector. This date-range selector lets you set the time period for which you want to analyze the reports. To set a new time period, click and drag the bubble-like endpoints on the date-range selector.

At the top of the date-range selector, select **Month**, **Week**, **Day**, **Hour**, or **Minutes** to see data by month, week, day, or hour.

The **Month** option displays six touch points, one for each of the last six months inclusive the current month.

The **Week** option displays one touch point for every week of a month.


The **Day** option displays one touch point for each day.

The **Hour** option displays 24 touch points, one for each hour of the day.

The **Minutes** option displays 48 touch points, one for every 30 minutes of the day.

9. At the top-right corner of the date-range selector, you find options to hide the date-range selector, view a grid representation of all your custom charts, and refresh the reports.

Click  to hide or unhide the date-range selector.

Click  to view a grid representation of all the charts that are a part of your custom view. Hovering over each chart gives you options to either remove the chart from your custom view or add the chart to your custom view.

Click  to refresh the reports with latest data from API calls.

Click to  delete a custom view.

Deleting a custom view deletes all its associated charts and data.

10. Each custom chart that you add to your custom view provides an action bar with options to edit and delete the chart.

Click  to edit the chart.

Click  to delete the chart.

#### Note

In any of the custom charts, if you choose `Line Chart` or `Stacked Bar Chart` chart types, the custom chart displays a time-wise trend of the report. For example, if you have a custom chart created for displaying the number of calls per API, then selecting the `Line Chart` type displays the number of calls based on the time period (Month, Week, or Day) you have selected.

## Related Information

[Find Your Way around Advanced API Analytics Dashboard \[page 706\]](#)

[Create Custom Dimensions and Measures \[page 714\]](#)


### 1.9.2.3 Create Custom Dimensions and Measures

Capture and analyze data using custom dimensions and custom measures.

#### Context

Advanced API Analytics provides a set of default dimensions and measures to track analytics data. However, if you need dimensions and measures that aren't included in the default list, you can create custom dimensions and measures.

With a custom dimension or a custom measure, you collect and analyze data that analytics don't automatically track. For instance, you want to capture API calls or API errors based on an API Key. Advanced API Analytics doesn't provide an out-of-the-box dimension that allows you to track data based on an API key. In such cases, you can define a custom dimension for capturing API-Key-based data. Similarly, you want to track the number of headers passed in an API call. In such cases, you can create a custom measure to track the total or average number of headers passed in an API call.

Perform the step-by-step instructions in this topic to create custom dimensions and measures. For visual instructions, you can also refer the following tutorial: [Create Custom Dimensions and Measures](#) 

## Procedure

1. In the analytics dashboard, choose *Custom Metric* from the *+Add* dropdown menu.
2. In the *Add Custom Metric* window, enter the name of the custom dimension or the custom measure that you want to add for tracking data. In this step, you enter just the names of custom dimensions and measures. However, for enabling data collection with them, you must reuse the names of custom dimensions or measures in the Statistics Collector policy of your API proxy. This procedure is explained in further steps.
3. Choose *OK*.
4. Choose the navigation icon on the left and choose **► Design ► APIs**.
5. From the APIs list, choose the required API for which you want to collect data using the custom metric.
6. Choose **► Policies ► Edit**.
7. Attach the [Statistics Collector Policy \[page 418\]](#) to the PreFlow of your ProxyEndpoint. For more information about how to add policies to API proxy, see [Policies \[page 205\]](#).
8. Open the payload of Statistics Collector policy that you attached to the API Proxy.

### Note

By default, the payload of Statistics Collector policy displays all the custom dimensions and measures that you've created. It displays them in a commented state with xml indicators `<!-- -->` as shown in the below sample payload:

### Sample Code

```
<!-- The policy collects data for each request and passes to the analytics
server. In the below sample payload, you can see a custom dimension
'APIKey' for
 collecting data based on API keys and a custom measure
'HeadersCount' for collecting the count of API headers passed in API
calls. -->
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<StatisticsCollector xmlns="http://www.sap.com/apimgmt">
 <Statistics>
 <!-- <Statistic name="APIKey" ref="request.header.APIKey"
type="string">999999</Statistic> -->
 <!-- <Statistic name="HeadersCount" ref="request.headers.count"
type="integer">0</Statistic> -->
 </Statistics>
</StatisticsCollector>
```

To enable data collection, you must uncomment the custom dimension or the measure with which you want to enable data tracking. In the below sample payload, data collection is enabled only for the custom dimension `APIKey`.

### Sample Code

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<StatisticsCollector xmlns="http://www.sap.com/apimgmt">
 <Statistics>
 <Statistic name="APIKey" ref="request.header.APIKey"
type="string">999999</Statistic>
 <!-- <Statistic name="HeadersCount" ref="request.headers.count"
type="integer">0</Statistic> -->
 </Statistics>
```

```
</StatisticsCollector>
```

9. After you've created the custom dimension or measure, navigate to the analytics dashboard. Add a custom view and create custom charts using the custom dimensions or measures you created.

#### Note

After creating a chart with custom dimension or custom measure, you'll experience a delay of 20-30 minutes before data starts appearing in the charts.

## Related Information

[Advanced API Analytics \[page 706\]](#)

[Create and Work with Custom Reports \[page 712\]](#)

## 1.9.2.4 Request to Modify Threshold Value for Charts

In the analytics dashboard, the values on the chart for a particular dimension are shown only up to a certain threshold. Any values beyond this threshold are grouped together and labelled as **Others**. By default, the threshold value for the charts is set to **25**.

### Context

To create a support ticket for changing the threshold value for charts, follow the steps described below.

### Procedure

1. Navigate to <https://launchpad.support.sap.com/>.
2. On the *SAP for Me* page, select the *Enter the SAP ONE Support Launchpad* tile.
3. On the *SAP ONE Support Launchpad*, select the *Report an Incident* tile.
4. In the *Create Incident* window, expand the *Description* category and enter the following details in the description field:
  - a. Specify the desired threshold value for charts.
  - b. Specify the *Subdomain Name* and *ID*.

To find the subdomain name and ID, follow these steps:

1. Open the SAP BTP Cockpit and navigate to your subaccount.
2. On the *Overview* page of your subaccount, locate the *Subdomain* attribute.
3. Make a note of the value marked against the *Subdomain* attribute. This value represents the subdomain name.

4. Additionally, note down the ID associated with your subaccount.
- c. Select the following component: **OPU-API-OD-OPS**
5. Submit the incident.

## 1.9.3 SAP Analytics Cloud for

SAP Analytics Cloud is an enterprise-wide solution that combines business intelligence, planning, and predictive analytics into one cloud environment. It provides a unified and secure public cloud experience to the users enabling faster decision making. For more information, see [SAP Analytics Cloud](#).

### Related Information

[Overview](#)

[Architecture and Abstract](#)

[Stories - API Management](#)

[Models](#)

[Data Connectivity](#)

## 1.10 Monetize APIs

provides monetization feature to all API providers to generate revenue for using the APIs.

As an API Admin, you can create a rate plan, attach it to a product in the , and publish the product in the API business hub enterprise. You can also view bill details of each developer in the .

As an application developer, in the API business hub enterprise you can create an application and add products to the application. Based on the product usage, you can view the corresponding bill details.

provides this feature through the following services:

- [Rate Plan Service \[page 718\]](#)
- [Billing Service \[page 723\]](#)

If you were creating, updating, or reading an application using the APIs and not through the user interface, then you need to switch to Subscription entity from Application entity to use the Monetize feature. For more information, see [Create or Update or Read an Application using Subscription key \[page 727\]](#)

## 1.10.1 Rate Plan Service

allows user to create rate plans and attach a rate plan to a product. Through a rate plan you can charge the application developers for the use of your APIs.

For more information see,

- [Create a Rate Plan \[page 718\]](#)
- [Attach Rate Plan to a Product \[page 720\]](#)
- [Update a Rate Plan \[page 721\]](#)
- [Delete a Rate Plan \[page 723\]](#)

### 1.10.1.1 Create a Rate Plan

Create a rate plan using the .

#### Prerequisites

You are assigned the admin role.

#### Context

You are creating a rate plan.

#### Procedure

1. Log on to the .
2. From the navigation bar, choose *Monetize*.
3. On the *Monetize* screen, choose *Create*.
4. On the *Create Rate Plan* screen, enter values for the following fields:
  - **Name:** Name of the rate plan.
  - **Description:** Outline of the plan.
  - **Frequency:** Monthly
  - **Currency:** Euro
  - **Basic Charge:** Minimum bill amount paid by the user after subscribing to the product associated with this rate plan.
  - **Rate per API Call:** Amount in Euros for one API call.

- **Plan Type:** Choose either *Basic* or *Tier*.
  - **Basic:** In Basic rate plan type, the rate charged per API call is fixed.
  - **Tier:** In Tier based rate plan type, the rate charged per API call varies based on the number of API calls.

#### Example

##### Tier based rate plan

Tier based rate plan

API Calls From	API Calls To	Rate per API Call
0	5000	0.0
5001	10000	0.5
10001		0.7

In the above example, for initial 5000 calls the rate charged is 0.0 per API call. For the next 5000 calls the rate charged is 0.5 per API call and for 10000 + calls the rate charged is 0.7 per API call. For instance, if 8000 calls are made, then the rates per API call is 0.0 for 0-5000 calls and 0.5 for the remaining 3000 calls.

#### Note

If the *API Calls To* field is left empty, then the system considers the field value to be unlimited.

5. Choose *Save*.

## Related Information

[Attach Rate Plan to a Product \[page 720\]](#)

[Update a Rate Plan \[page 721\]](#)

[Delete a Rate Plan \[page 723\]](#)

## 1.10.1.2 Attach Rate Plan to a Product

Attach a rate plan to a product using the .

### Prerequisites

- You are assigned the admin role.
- You have created a rate plan in the .

#### ⓘ Note

You can only attach rate plans to those products that do not have any rate plans associated with them. A product can only be associated with one rate plan. you can also attach a rate plan to a product during the product creation.

#### ⓘ Note

If you try changing a rate plan or add a new rate plan to a product, all the existing applications of this product will remain unaffected by the changes. For example, if you add a rate plan to a product associated with the application, which has already been subscribed, this will not impact the current billing of the application.

### Context

You are attaching a rate plan to a product.

### Procedure

1. Log on to the .
2. Choose the navigation icon on the left and choose **Design > APIs**.
3. On the **Design > APIs** page, choose **Products**.
4. From the list of products available, select the product to which you want to add the rate plan.
5. On the Product details screen, choose **RATE PLAN**.
6. Choose **Edit → Add Plan**.
7. In the **Add Rate Plan** window, select the required rate plan from the list of available rate plans.  
You can click an individual rate plan to view the description and details of that particular rate plan.
8. Choose **OK**.

Rate plan will not be applicable to existing applications associated with the product to which the rate plan is attached. However, the rate plan will be applicable, if a new application uses the product to which the rate plan is attached.



## Related Information

[Create a Rate Plan \[page 718\]](#)

[Update a Rate Plan \[page 721\]](#)

[Delete a Rate Plan \[page 723\]](#)

### 1.10.1.3 Update a Rate Plan

Update a rate plan using the .

#### Prerequisites

You are assigned the admin role.

#### Context

You are updating a rate plan.

#### Procedure

1. Log on to the .
2. From the navigation bar, choose *Monetize*.
3. On the *Monetize* screen, choose *RATE PLANS*.
4. On the *RATE PLANS* screen, select the rate plan that you want to update.
5. Choose *Edit*.

You can update the following fields only :

- **Name:** Name of the rate plan.
- **Description:** Outline of the plan.
- **Frequency:** Monthly
- **Currency:** Euro
- **Basic Charge:** Minimum bill amount paid by the user after subscribing to the product associated with this rate plan.
- **Rate per API Call:** Amount in euros for one API call.
- **Plan Type:** Choose either *Basic* or *Tier*.
  - **Basic:** In Basic rate plan type, the rate charged per API call is fixed.

- **Tier:** In Tier based rate plan type, the rate charged per API call varies based on the number of API calls.

### Example

Tier based rate plan

Tier based rate plan

API Calls From	API Calls To	Rate per API Call
0	5000	0.0
5001	10000	0.5
10001		0.7

In the above example, for initial 5000 calls the rate charged is 0.0 per API call. For the next 5000 calls the rate charged is 0.5 per API call and for 10000 + calls the rate charged is 0.7 per API call. For instance, if 8000 calls are made, then the rates per API call is 0.0 for 0-5000 calls and 0.5 for the remaining 3000 calls.

### Note

If the *API Call To* field is left empty, then the system considers the field value to be unlimited.

6. Choose *Save*.

### Note

Updated rate plan is applicable for new subscriptions only.

## Related Information

[Create a Rate Plan \[page 718\]](#)

[Update a Rate Plan \[page 721\]](#)

[Delete a Rate Plan \[page 723\]](#)

## 1.10.1.4 Delete a Rate Plan

Delete a rate plan using the .


### Prerequisites

You are assigned the admin role.

### Context

You are deleting a rate plan.

### Procedure

1. Log on to the .
2. From the navigation bar, choose *Monetize*.
3. On the *Monetize* screen, choose *RATE PLANS*.
4. in the *Actions* column, choose  to delete a rate plan.
5. Choose *Yes*.

#### Note

Deleted rate plan is not available for new subscriptions, but is available for existing subscriptions.

### Related Information

[Create a Rate Plan \[page 718\]](#)

[Update a Rate Plan \[page 721\]](#)

[Attach Rate Plan to a Product \[page 720\]](#)

## 1.10.2 Billing Service

Billing service is available in both and API business hub enterprise.

Using billing service, you can view the bill details and download bill details for a specific developer and for a specific month.

Service to view bills :

- URL: `https://<consumer API-Portal host>:<port>//api/1.0/apimgmt/monetize/bills`
- The following table describes the query parameters required to view the bill details.

Query parameters

Parameter name	Required in API portal	Required in API business hub enterprise	Description	Example
Month	Yes	Yes	Month in MM format	Month = 03
Year	Yes	Yes	Year in YYYY format	Year = 2017
developer_id	Yes	No	Developer e-mail Id	developer_id = jon.doe@sap.com
application_id	No	No	Id of a specific application for which bill has to be generated	application_id = 6C7F88BB-74BE-4CC C- A49A-6A8F2BF1EAC1

- You can also view the bill details in the and API business hub enterprise. For more information see,
  - [View Bill Details in the \[page 724\]](#)
  - [View Bill Details in the API business hub enterprise \[page 725\]](#)

## 1.10.2.1 View Bill Details in the

View bill details in thefor all the applications and products assigned to a particular developer.

### Prerequisites

You are assigned the admin role.

### Procedure

1. Log on to .
2. From the navigation bar, choose *Monetize*.
3. On the *Monetize* screen, choose *BILLS*.
4. Select the billing month from the *Month* and *Year* dropdown boxes.  
By default, the bill details for the current month are displayed.

5. From the list of developers, select the developer you want to view the bill details for.

The *Bill Detail* window shows a list of applications the developer is subscribed to and the corresponding bill amount for each application. You can view the list of products assigned to each application by clicking the application.

## 1.10.2.2 View Bill Details in the API business hub enterprise

View the bill details in the API business hub enterprise for all the applications subscribed by a developer.

### Prerequisites

You are assigned the application developer role.

### Procedure

1. Log on to the API business hub enterprise.
2. Navigate to, *My Workspace*.
3. The *Cost* section displays the billing details for all the application subscribed by the developer in two charts:
  - *Aggregated Costs in Euros*: displays the bill details for the last six months.
  - Cost for selected month: displays the cost for the selected month. By default, the cost for the current month is displayed.

To view the cost for a specified application. Navigate to the required application details screen, by choosing the required application. Cost pertaining to that application is visible in the *Cost* section in the detailed application screen.

## 1.10.2.3 Download Bill Details from

Download bill details using the .

### Prerequisites

You are assigned the admin role.

## Context

To download the bill details, proceed as follows:


## Procedure

1. Log on to .
2. From the navigation bar, choose *Monetize*.
3. On the *Monetize* screen, choose *BILLS*.
4. Select the billing month from the *Month* and *Year* dropdown boxes.

By default, the bill details for the current month is displayed.

### Note

You cannot download the bill details for the current month.

5. In the Actions column, choose  icon to download the bill details.  
Alternatively, you can download the bill details by choosing the *Total Bill Amount* for a developer. Choose *Download* in the *Bill Detail* window.

The bill details are downloaded in a .csv file format.

In the .csv file generated, leading = signs that might have been present in any user input (such Product Title, Application Title, or Rateplan Title) are trimmed.

## 1.10.2.4 Download Bill Details from API business hub enterprise

Download bill details using the API business hub enterprise.

## Prerequisites

You are assigned the application developer role.

## Context

To download the bill details, proceed as follows:

## Procedure

1. Log on to the API business hub enterprise.
2. Navigate to, [My Workspace](#).
3. In the [Cost](#) section, choose the billing month from the [Aggregated Costs in Euros](#) chart.  
By default, the cost for the current month is displayed.

### Note

You cannot download the bill details for the current month.

4. Choose [Download](#) action item.

The bill details are downloaded in a .csv file format.

In the .csv file generated, leading = signs that might have been present in any user input (such Product Title, Application Title, or Rateplan Title) are trimmed.

## 1.10.3 Create or Update or Read an Application using Subscription key

Creating, updating, and reading an application using the Subscription key.

You can use the following metadata for Subscription entity.

### Note

To use the monetization feature, it is recommended to use Subscription entity and not the Application entity to create or update or read an application.

### Sample Code

```
<EntityType Name="SubscriptionsType">
 <Key>
 <PropertyRef Name="id"/>
 </Key>
 <Property Name="id" Type="Edm.String" Nullable="false" MaxLength="256"/>
 <Property Name="reg_id" Type="Edm.String" MaxLength="256"/>
 <Property Name="app_id" Type="Edm.String" MaxLength="256"/>
 <Property Name="product_id" Type="Edm.String" MaxLength="256"/>
 <Property Name="developer_id" Type="Edm.String" MaxLength="256"/>
 <Property Name="ratePlan_id" Type="Edm.String" MaxLength="256"/>
 <Property Name="validFrom" Type="Edm.DateTime"/>
 <Property Name="validTo" Type="Edm.DateTime"/>
 <Property Name="app_name" Type="Edm.String" MaxLength="255"/>
 <Property Name="isSubscribed" Type="Edm.Boolean"/>
 <Property Name="status" Type="Edm.String" MaxLength="255"/>
 <Property Name="comment" Type="Edm.String" MaxLength="2048"/>
 <Property Name="created_by" Type="Edm.String" MaxLength="255"/>
 <Property Name="created_at" Type="Edm.DateTime"/>
 <Property Name="modified_by" Type="Edm.String" MaxLength="255"/>
 <Property Name="modified_at" Type="Edm.DateTime"/>
 <NavigationProperty Name="ToApplication"
 Relationship="developer.Subscriptions_ApplicationsType"
 FromRole="SubscriptionsDependent" ToRole="ApplicationsDependent"/>
</EntityType>
```

```

<NavigationProperty Name="ToRatePlan"
Relationship="developer.Subscriptions_RatePlansType"
FromRole="SubscriptionsDependent" ToRole="RatePlansDependent"/>
<NavigationProperty Name="ToAPIProduct"
Relationship="developer.Subscriptions_APIProductsType"
FromRole="SubscriptionsDependent" ToRole="APIProductsPrincipal"/>
</EntityType>

```

URL of Subscription Entity: <developer portal base url>/odata/1.0/data.svc/APIMgmt.Subscriptions

Use the below mentioned payload to create an application using Subscription entity :

URL: <developer portal base url>/odata/1.0/data.svc//APIMgmt.Applications

Request Method: POST

Content-Type: application/json

### Code Syntax

#### Payload

```

{
 "id": "00000000000000000000000000000000",
 "version": "1",
 "title": "App_Title",
 "description": "Description",
 "callbackurl": "http://www.callbackurl.com",
 "ToSubscriptions": [
 {
 "id": "00000000000000000000000000000000",
 "ToAPIProduct": [
 {
 "__metadata": {
 "uri": "APIMgmt.APIProducts('Product_Catalog')"
 }
 }
],
 "ToRatePlan": [
 {
 "__metadata": {
 "uri": "APIMgmt.RatePlans('E8BF82AA-F7B0-427F-881A-D246A047BBD0')"
 }
 }
]
 }
]
}

```

Use the below mentioned payload to update fields like title or callback url or description in the application using Subscription entity :

URL: <developer portal base url>/odata/1.0/data.svc/\$batch

Method: POST

Content-Type: multipart/mixed; boundary=batch\_349d851f-79ed-44bc-b67a-3159f7cfcc17

Content-Length: <length of the content>



## Code Syntax

### Payload

```
--batch_b72a-e938-270d
Content-Type: multipart/mixed; boundary=changeset_319c-d23e-258e
--changeset_319c-d23e-258e
Content-Type: application/http
Content-Transfer-Encoding: binary

PUT APIMgmt.Applications('238D7CA1-3F61-470B-BC73-37FF311739E2') HTTP/1.1
Accept-Language: en-US
Accept: application/json
MaxDataServiceVersion: 2.0
DataServiceVersion: 2.0
Content-Type: application/json
Content-Length: 355
{"id":"238D7CA1-3F61-470B-BC73-37FF311739E2","title":"App_Title_Updated","callbackurl":"http://www.callbackurl_updated.com","description":"Description Updated. ","app_key":"FuoGpWnZR0SJJedimgRpACH6XaiJc1XR","reg_id":"5f1007f1cd5c47ea8a209ce1056798f8","version":"1","app_secret":"rpZAJyTgwFGDLyeh","valid_from":null,"valid_to":null,"developer_id":"I305297"}
--changeset_319c-d23e-258e--
--batch_b72a-e938-270d--
```

Use the below mentioned payload to update the application to add and remove a product using Subscription entity :

URL: <developer portal base url>/odata/1.0/data.svc/\$batch

Method: POST

Content-Type: multipart/mixed; boundary=batch\_349d851f-79ed-44bc-b67a-3159f7cfcc17

Content-Length: <length of the content>

## Code Syntax

### Payload

```
--batch_ce8f-d810-b289
Content-Type: multipart/mixed; boundary=changeset_0750-94e8-367a
--changeset_0750-94e8-367a
Content-Type: application/http
Content-Transfer-Encoding: binary
PUT APIMgmt.Applications('238D7CA1-3F61-470B-BC73-37FF311739E2') HTTP/1.1
RequestId: cf1da741-bd68-401e-95d7-9bcf0475112b
Accept-Language: en-US
Accept: application/json
MaxDataServiceVersion: 2.0
DataServiceVersion: 2.0
x-csrf-token: CF4601D494334B00239A025C270BDBF1
Content-Type: application/json
Content-Length: 355
{"id":"238D7CA1-3F61-470B-BC73-37FF311739E2","title":"App_Title_Updated","callbackurl":"http://www.callbackurl_updated.com","description":"Description Updated. ","app_key":"FuoGpWnZR0SJJedimgRpACH6XaiJc1XR","reg_id":"5f1007f1cd5c47ea8a209ce1056798f8","version":"1","app_secret":"rpZAJyTgwFGDLyeh","valid_from":null,"valid_to":null,"developer_id":"I305297"}
--changeset_0750-94e8-367a
Content-Type: application/http
Content-Transfer-Encoding: binary
POST APIMgmt.Subscriptions HTTP/1.1
```

```

RequestId: cf1da741-bd68-401e-95d7-9bcf0475112b
Accept-Language: en-US
Accept: application/json
MaxDataServiceVersion: 2.0
DataServiceVersion: 2.0
x-csrf-token: CF4601D494334B00239A025C270BDBF1
Content-Type: application/json
Content-Length: 322
{"id":"00000000000000000000000000000000","ToAPIProduct":[{"__metadata":{"uri":"APIMgmt.APIProducts('NorthwindProd')}}],"ToApplication":[{"__metadata":{"uri":"APIMgmt.Applications('238D7CA1-3F61-470B-BC73-37FF311739E2')}}],"ToRatePlan":[{"__metadata":{"uri":"APIMgmt.RatePlans('1D8F672C-DFDF-4A73-8BB7-63E649C6BB57')}}]}
--changeset_0750-94e8-367a
Content-Type: application/http
Content-Transfer-Encoding: binary
PUT APIMgmt.Subscriptions('3651b33d5fc34f3aa72df7fc0c529450') HTTP/1.1
RequestId: cf1da741-bd68-401e-95d7-9bcf0475112b
Accept-Language: en-US
Accept: application/json
MaxDataServiceVersion: 2.0
DataServiceVersion: 2.0
x-csrf-token: CF4601D494334B00239A025C270BDBF1
Content-Type: application/json
Content-Length: 449
{"id":"3651b33d5fc34f3aa72df7fc0c529450","isSubscribed":false,"app_id":"238D7CA1-3F61-470B-BC73-37FF311739E2","product_id":"Product_Catalog","developer_id":"I305297","ToAPIProduct":[{"__metadata":{"uri":"APIMgmt.APIProducts('Product_Catalog')}}],"ToRatePlan":[{"__metadata":{"uri":"APIMgmt.RatePlans('E8BF82AA-F7B0-427F-881A-D246A047BBD0')}}],"ToApplication":[{"__metadata":{"uri":"APIMgmt.Applications('238D7CA1-3F61-470B-BC73-37FF311739E2')}}]}
--changeset_0750-94e8-367a--
--batch_ce8f-d810-b289--

```

Use the below mentioned url and method to delete an application :

URL: <developer portal base url>/odata/1.0/data.svc/APIMgmt.Applications('<app\_id>')

Method: DELETE

Use the below mentioned url and method to read all applications using Subscription entity :

URL:<developer portal base url> /odata/1.0/data.svc/APIMgmt.Applications

Method: GET

Response : It will fetch only application attributes like title, call back url, description.

- It will not fetch app key and secret.
- If the application is created using Subscription Entity, then the attached products will not be shown here.
- Although, if an application is created using older APIs (only using Application Entity and Application to Product linkage), then the attached product details will be shown in the navigation "ToAPIProductsDetails".

Use the below mentioned url and method to read a specific applications using Subscription entity :

URL: <developer portal base url>/odata/1.0/data.svc/APIMgmt.Applications('<app\_id>')?

\$expand=ToAPIProductsDetails,ToSubscriptions/ToAPIProduct,ToSubscriptions/ToRatePlan&\$format=json

Method: GET

Response : This API will fetch all the application related details.

- If the application is created using Subscription entity, the associated products will be found in the navigation "ToSubscriptions/ToAPIProduct".
- If the application is created using older API (only using Application Entity and Application to Product linkage), the attached product details will be found in the navigation property "ToAPIProductsDetails".

#### 📌 Note

- If a user is not willing to use the Monetization feature, then he or she may continue using Application's service.
- If a user is willing to use Monetization feature, then he or she must switch to Subscription entity.
- In order to use Monetization, the user needs to create a new rate plan and a new product. Attach the rate plan to the product, publish the product, and let the application developer consume the product via a new application created using Subscription entity.
- It is not recommended to use mix and match of Application entity services and Subscription Entity.
- It is not recommended to use the application created using Application Entity's service for Monetization purpose.
- It is not recommended to use the application created using Application Entity Service to add a product having rate plan.
- If a user has applications created using Application's entity service and applications created using Subscriptions entity, then user must make two calls while reading the applications:
  - One read call on Application's entity.
  - One read call on Subscription's entity.

## 1.11 Discover API Packages

API packages supported by the API Management are available in SAP Business Accelerator Hub. You can discover these API packages on .

### Discover

Log on to the , and choose ► [Discover](#) ► [APIs](#) ► to explore the packages under the following sections:

- **Highlights:** Showcased packages are displayed in the *Featured* section. Recently published packages are displayed in the *Latest* section.
- **All:** This section lists all the packages in the portal.

Each package is displayed as an individual tile, along with the name of the package, the rating of the package, and a brief description of the package.

If the content has been developed by a partner, the package also displays a **Partner** label.

To find out more about the features of the packages on the , see [Package Details \[page 732\]](#).

## 1.11.1 Package Details

A package is a container that can hold different types of content. It typically contains APIs and policy templates. It can also contain documentation and links.

Each package has the following details:

- **Overview**

The [Overview](#) provides the following information about the package:

- **Description:** Details of the package and the scenarios it can be used for.
- **Supported Platform:** API Management
- **Category:** APIs
- **Created on:** The date and time the package was created.

- **Artifacts**

The [Artifacts](#) section displays the APIs and policy templates available in the package.

You can view the details of an API, a policy template, or an API proxy by choosing the respective artifact. This opens a screen that provides the details of that particular artifact.

To copy APIs, choose the [Action](#) icon and choose [Copy](#). In the [Copy API](#) window, provide the necessary information. You can either leave the information displayed for [API Details](#) as it is, or you can change it.

After the action is completed, you can view the copied API by navigating to [Design > APIs > POLICY TEMPLATES](#). Alternatively, you can copy the APIs by choosing the respective artifact and then, on the screen that opens, choosing [Copy](#).

To copy a policy template, choose the [Action](#) icon and select [Copy](#). After the action is completed, you can view the copied policy template by navigating to [Design > APIs > POLICY TEMPLATES](#). Alternatively, you can copy the policy template by choosing the respective artifact and then, on the screen that opens, choosing [Copy](#).

- **Documents**

This section contains any documents associated with the package.

- **Tags**

Country, Product, Keyword, Lines of Business, Industry, and other tags for the package are displayed here.

- **Ratings**

This section contains user ratings and feedback for the package.

- **View in SAP Business Accelerator Hub**

At the package level, you can view and copy APIs to the . If you want to perform actions such as trying out the API or generating the code, then navigate to SAP Business Accelerator Hub by choosing [View in SAP Business Accelerator Hub](#). Choosing the link takes you to the same package in SAP Business Accelerator Hub.

## 1.12 API Documentation

This section contains additional instructions on how to effectively use and integrate with an API.

The standard documentation for API Management APIs is already available on the [SAP Business Accelerator Hub](#).

It provides you with a concise reference manual containing additional information required to work with various entities in the API portal. For example, when to expect a \$batch call, the format of a \$batch call, information

about expected headers, payload format, how to create/update single and multiple records, mandatory and optional fields, and, so on.

## 1.13 Security

This section describes how to secure API Management applications.

Section	Description
User Authentication	To access API Management application one needs to have a valid SCN user registered with API Management solution. The SCN credentials should be used for logon to API Management solution.
Authorization	In API Management, you provide authorization to users by assigning relevant roles. For more information on how to provide authorizations, see .
Securing APIs	Secure your APIs by referring to the following security policies supported by API Management: <ul style="list-style-type: none"><li>• <a href="#">Basic Authentication [page 253]</a></li><li>• <a href="#">OAuth v2.0 [page 355]</a></li><li>• <a href="#">OAuth v2.0 GET [page 368]</a></li><li>• <a href="#">OAuth v2.0 SET [page 371]</a></li><li>• <a href="#">Verify API Key [page 383]</a></li><li>• <a href="#">SAML Assertion Policy [page 376]</a></li></ul>
Security Best Practices	Security policies provide information on how to control access to your APIs with OAuth, API key and other threat protection features. For more information, refer to the following policies supported by API Management: <ul style="list-style-type: none"><li>• <a href="#">Access Control [page 207]</a></li><li>• <a href="#">JSON Threat Protection [page 297]</a></li><li>• <a href="#">XML Threat Protection [page 401]</a></li><li>• <a href="#">Message Validation Policy [page 380]</a></li><li>• <a href="#">Regular Expression Protection [page 414]</a></li></ul>
Traffic management	API Management supports the following traffic management policies: <ul style="list-style-type: none"><li>• <a href="#">Quota [page 336]</a></li><li>• <a href="#">Spike Arrest [page 353]</a></li><li>• <a href="#">Concurrent Rate Limit [page 255]</a></li></ul>

For more information on Security policies, see <https://blogs.sap.com/2017/08/22/sap-cloud-platform-api-management-api-security-best-practices/>

## 1.13.1 Data Protection and Privacy for API Management

### Glossary

Term	Definition
<b>Blocking</b>	A method of restricting access to data for which the primary business purpose has ended.
<b>Business purpose</b>	A legal, contractual, or otherwise justified reason for the processing of personal data. The assumption is that any purpose has an end that is usually already defined when the purpose starts.
<b>Consent</b>	The action of the data subject confirming that the usage of his or her personal data shall be allowed for a given purpose. A consent functionality allows the storage of a consent record in relation to a specific purpose and shows if a data subject has granted, withdrawn, or denied consent.
<b>Deletion</b>	Deletion of <b>personal data</b> so that the data is no longer available.
<b>End of business</b>	Date where the business with a data subject ends, for example the order is completed, the subscription is canceled, or the last bill is settled.
<b>End of purpose (EoP)</b>	End of purpose and start of blocking period. The point in time, when the primary processing purpose ends (e.g. contract is fulfilled).
<b>End of purpose (EoP) check</b>	A method of identifying the point in time for a data set when the processing of <b>personal data</b> is no longer required for the primary <b>business purpose</b> . After the <b>EoP</b> has been reached, the data is <b>blocked</b> and can only be accessed by users with special authorization (for example, tax auditors).

Term	Definition
<b>Personal data</b>	<p>Any information relating to an identified or identifiable natural person ("data subject"). An identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or one or more factors specific to the physical, physiological, genetic, mental, economic, cultural, or social identity of that natural person.</p> <p>Any information relating to the application developer or the Cloud Foundry developer using API Management service.</p>
<b>Residence period</b>	<p>The period of time between the end of business and the end of purpose (EoP) for a data set during which the data remains in the database and can be used in case of subsequent processes related to the original purpose. At the end of the longest configured residence period, the data is blocked or deleted. The residence period is part of the overall retention period.</p>
<b>Retention period</b>	<p>The period of time between the end of the last business activity involving a specific object (for example, a business partner) and the deletion of the corresponding data, subject to applicable laws. The retention period is a combination of the residence period and the blocking period. API Management has a retention period of six months and all the data is automatically cleaned up after the retention period.</p>
<b>Referenced data</b>	<p>Any information relating to application developer's application.</p>

## User Consent

Various types of customer data are processed by and stored on API Management at different times. This data gets the highest level of protection, and SAP takes dedicated measures to guarantee this security level.

To comply with user consent, SAP customers should customize the API Management to use their own identity provider. SAP customers using the custom identity provider should ensure that the necessary mechanism for user consent is available to allow personal data (of a natural person such as a customer, contact, or account) to be collected as well as transferred to the solution.

## Read-Access Logging

Read Access Logging (RAL) is used to monitor and log read access to sensitive data. Data may be categorized as sensitive by law, by external company policy, or by internal company policy.

API Management does not store any sensitive personal data.

## Information Report

The only personal data of data subjects stored in API Management is the user ID. This user ID is stored whenever a user creates an API artifact, for example an API proxy or API product, and this user ID can be obtained (read) only from that artifact.

To enable data subjects to obtain information about their personal data in the API Business Hub Enterprise, we provide the following service to retrieve their personal information:

- [Service to View User Details on API business hub enterprise \[page 737\]](#)

## Erasure

When handling personal data, consider the legislation in the different countries where your organization operates. After the data has passed the end of purpose, regulations may require you to delete the data. However, additional regulations may require you to keep the data longer. During this period you must block access to the data by unauthorized persons until the end of the six months retention period, when the data is finally deleted.

Personal data can also include referenced data. The challenge for deletion and blocking is to first handle referenced data and then other data, such as business partner data.

API Management stores the API portal administrator's user ID. Storing the user ID is a business requirement and the user ID is deleted when the resources created by the API portal administrator are removed.

API Management stores personal information such as first name, last name, user ID, and e-mail ID of users who have logged on to the Developer Portal. All the personal information stored in the application is deleted when the access for the corresponding user is revoked.

In API Management, application developers can contact their developer portal administrators to have their personal data erased and access revoked. For more information, see [Revoke Access \[Classic Design\] \[page 668\]](#) and [Delete Data of Unregistered Users \[page 670\]](#).

## Change Log

For auditing purposes or for legal requirements, changes made to personal data should be logged, making it possible to monitor who made changes and when.

API Management does not allow users to make any changes to their personal data via the API Management application. However, changes made in the identity provider are synced to the API Management application and the sync operation is logged by API Management. Changes made in the identity provider should be logged by the identity provider.



### 1.13.1.1 Service to View User Details on API business hub enterprise

API Management allows user to view their personal data stored in the API business hub enterprise.

**Service to view personal data on API business hub enterprise:**

- URL: `https://<Dev-Portal-URL>/api/1.0/user`
- Method: GET
- Response: Fetches your personal details stored in API business hub enterprise.

#### Sample Code

Sample response

```
{
 "Name": "<user ID>",
 "FirstName": "<First name>",
 "LastName": "<Last name>",
 "LoggedOut": false,
 "Email": "<e-mail id>"
}
```

### 1.13.1.2 Auditing and Logging Information for API Management

Here you can find a list of the security events that are logged by TECHNICAL COMPONENT.

Security events written in audit logs

Event grouping	What events are logged	How to identify related log events	Additional information
API Proxy	Create API Proxy	<ul style="list-style-type: none"><li>• action: create</li><li>• objectType: PROXY</li></ul> <p data-bbox="804 501 900 526">Audit Log</p> <p data-bbox="804 548 900 573">Example:</p> <div data-bbox="804 595 1091 1547" style="border: 1px solid #ccc; padding: 10px;"><p data-bbox="820 607 1038 631">↔ Output Code</p><pre data-bbox="836 674 1054 1514">{"\\"uuid\\":\\"&lt;msg GUID&gt;\\",\\"user \\":\\"&lt;user e- mail&gt;\\", \\"time\\":\\"202 1-06-09T10:34: 32.369Z\\", \\"id\\":\\"b370c 7ed-a995-4ac8- b207-9ba59f470 e55\\", \\"success\\":tr ue,\\"object\\": {\\"type\\":\\"PR OXY\\", \\"id\\": {\\"class\\":\\"c om.sap.apimgmt .asmprov.core. processor.APIP roxyProcessor\\ ", \\"objectId\\":\\" PROXY\\",\\"act ion\\":\\"create \\", \\"message\\":\\" &lt;based on the event the corresponding message will be logged&gt;}"</pre></div>	<a href="#">Different Methods of Creating an API Proxy [page 477]</a>

Event grouping	What events are logged	How to identify related log events	Additional information
API Proxy	Update API Proxy	<ul style="list-style-type: none"> <li>• action: update</li> <li>• objectType: PROXY</li> </ul> <p>Audit Log</p> <p>Example:</p>	<a href="#">Edit an API Proxy [page 585]</a>

↔ Output Code

```

{"\ "uuid\ ": "\ <
msg
GUID> \ ", "\ user
\ ": "\ <user e-
mail> \ ",
\ "time\ ": "\ 202
1-06-09T10:34:
32.369Z \ ",
\ "id\ ": "\ <ID> \
",
\ "success\ ": tr
ue, \ "object\ ":
{ \ "type\ ": "\ PR
OXY \ ",
\ "id\ ":
{ \ "class\ ": "\ c
om.sap.apimgmt
.asmprov.core.
processor.APIP
roxyProcessor \
",
\ "objectId\ ": \
"PROXY \ ", \ "act
ion\ ": "\ update
\ ",
\ "message\ ": "\
<based on the
event the
corresponding
meassage will
be logged> }"

```

Event grouping	What events are logged	How to identify related log events	Additional information
API Proxy	Delete API Proxy	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: PROXY</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```

{"\ "uuid\ ": "\ <
msg
GUID> \ ", "\ user
\ ": "\ <user e-
mail> \ ",
\ "time\ ": "\ 202
1-06-09T10:34:
32.369Z \ ",
\ "id\ ": "\ <ID> \
"
\ "success\ ": tr
ue, \ "object\ ":
{ \ "type\ ": "\ PR
OXY \ ",
\ "id\ ":
{ \ "class\ ": "\ c
om.sap.apimgmt
.asmprov.core.
processor.APIP
roxyProcessor \
",
\ "objectId\ ": \
"PROXY \ ", \ "act
ion\ ": "\ delete
\ ",
\ "message\ ": "\
<based on the
event the
corresponding
message will
be logged> }"

```

Event grouping	What events are logged	How to identify related log events	Additional information
API Product	Create API Product	<ul style="list-style-type: none"> <li>• action: create</li> <li>• objectType: APIPRODUCT</li> </ul> <p>Audit Log</p> <p>Example:</p>	<a href="#">Create a Product [page 653]</a>

#### ↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
\"success\":tr
ue,\"object\":
{\"type\":\"AP
IPRODUCT\",
\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.core.
processor.APIP
roductProcesso
r\",
\"objectId\":\
\"APIPRODUCT\",
\"action\":\"c
reate\",
\"message\":\
<based on the
event the
corresponding
meassage will
be logged>}"
```

Event grouping	What events are logged	How to identify related log events	Additional information
API Product	Update API Product	<ul style="list-style-type: none"> <li>• action: update</li> <li>• objectType: APIPRODUCT</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```

{"\uid\":"<msg
GUID>","\user
\":"<user e-
mail>","\,
"time\":"202
1-06-09T10:34:
32.369Z","\,
"id\":"<ID>\"
\"success\":"tr
ue,\"object\":"
{\"type\":"API
PRODUCT\" ,
\"id\":"
{\"class\":"c
om.sap.apimgmt
.asmprov.core.
processor.APIP
roductProcesso
r\" ,
\"objectId\":"
\"APIPRODUCT\" ,
\"action\":"\"u
pdate\" ,
\"message\":"\"
<based on the
event the
corresponding
meassage will
be logged>\"}

```

Event grouping	What events are logged	How to identify related log events	Additional information
API Product	Delete API Product	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: APIPRODUCT</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
\"success\":tr
ue,\"object\":
{\"type\":\"AP
IPRODUCT\",
\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.core.
processor.APIP
roductProcesso
r\",
\"objectId\":\
\"APIPRODUCT\",
\"action\":\"d
elete\",
\"message\":\
<based on the
event the
corresponding
meassage will
be logged>}"
```

Event grouping	What events are logged	How to identify related log events	Additional information
Application	Create Application Name	<ul style="list-style-type: none"> <li>action: create</li> <li>objectType: APPLICATION</li> </ul> <p>Audit Log</p> <p>Example:</p>	<a href="#">Create an Application [Classic Design] [page 686]</a>

#### ↔ Output Code

```

{"\uid\":"<msg
GUID>","\user
\":"<user e-
mail>","\,
"time\":"202
1-06-09T10:34:
32.369Z","\,
"id\":"<ID>\"
\"success\":"tr
ue,\"object\":"
{\"type\":"AP
PLICATION\" ,
\"id\":"
{\"class\":"c
om.sap.apimgmt
.asmprov.core.
processor.Appl
icationProcess
or\" ,
\"objectId\":"
\"APPLICATION\"
,\"action\":"
create\" ,
\"message\":"
<based on the
event the
corresponding
meassage will
be logged>}"

```



Event grouping	What events are logged	How to identify related log events	Additional information
Application	Update Application Name	<ul style="list-style-type: none"> <li>• action: update</li> <li>• objectType: APPLICATION</li> </ul> Audit Log Example:	

↔ Output Code

```

{"\ "uuid\ ": "\ <
msg
GUID> \", \ "user
\ ": \ "<user e-
mail> \",
\ "time\ ": \ "202
1-06-09T10:34:
32.369Z \",
\ "id\ ": \ "<ID> \
"
\ "success\ ": tr
ue, \ "object\ ":
{ \ "type\ ": \ "AP
PLICATION \",
\ "id\ ":
{ \ "class\ ": \ "c
om.sap.apimgmt
.asmprov.core.
processor.Appl
icationProcess
or \",
\ "objectId\ ": \
"APPLICATION \",
\ "action\ ": \ "
update \",
\ "message\ ": \ "
<based on the
event the
corresponding
meassage will
be logged> }"

```

Event grouping	What events are logged	How to identify related log events	Additional information
Application	Delete Application Name	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: APPLICATION</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```

{"\ "uuid\ ":\" <
msg
GUID>\",\"user
\ ":\" <user e-
mail>\",
\"time\ ":\"202
1-06-09T10:34:
32.369Z\",
\"id\ ":\" <ID>\
\"
\"success\ ":tr
ue,\"object\ ":
{\ "type\ ":\"AP
PLICATION\",
\"id\ ":
{\ "class\ ":\"c
om.sap.apimgmt
.asmprov.core.
processor.Appl
icationProcess
or\",
\"objectId\ ":\"
APPLICATION\",
\"action\ ":\"
delete\",
\"message\ ":\"
<based on the
event the
corresponding
meassage will
be logged>}"

```

Event grouping	What events are logged	How to identify related log events	Additional information
API Provider	Create API Provider	<ul style="list-style-type: none"> <li>• action: create</li> <li>• objectType: APIPROVIDER</li> </ul> <p>Audit Log</p> <p>Example:</p>	<a href="#">Create an API Provider [page 538]</a>

#### ↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>"
"success":tr
ue,"object":
{"type":"AP
IPROVIDER",
"id":
{"class":"c
om.sap.apimgt
.asmprov.core.
processor.APIP
roviderProcess
or",
"objectId":"
APIPROVIDER",
"action":"
create",
"message":"
<based on the
event the
corresponding
meassage will
be logged>}"
```

Event grouping	What events are logged	How to identify related log events	Additional information
API Provider	Update API Provider	<ul style="list-style-type: none"> <li>• action: update</li> <li>• objectType: APIPROVIDER</li> </ul> <p>Audit Log:</p> <p>Example:</p>	

↔ Output Code

```

{"\uid\":"<msg
GUID>","\user
\":"<user e-
mail>","\,
"time\":"202
1-06-09T10:34:
32.369Z","\,
"id\":"<ID>\"
\"success\":tr
ue,\"object\":
{\"type\":"API
PROVIDER\",
\"id\":
{\"class\":"c
om.sap.apimgmt
.asmprov.core.
processor.APIP
roviderProcess
or\",
\"objectId\":"
APIPROVIDER\"
,\"action\":"
update\",
\"message\":"
<based on the
event the
corresponding
meassage will
be logged>}"}

```

Event grouping	What events are logged	How to identify related log events	Additional information
API Provider	Delete API Provider	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: APIPROVIDER</li> </ul> <p>Audit Log:</p> <p>Example:</p>	

#### ↔ Output Code

```
"{"uuid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"b370c
7ed-a995-4ac8-
b207-9ba59f470
e55","
"success":tr
ue,"object":
{"type":"API
PROVIDER",
"id":
{"class":"c
om.sap.apimgmt
.asmprov.core.
processor.APIP
roviderProcess
or",
"objectId":"
APIPROVIDER",
"action":"de
lete",
"message":"
<based on the
event the
corresponding
meassage will
be logged>"}"
```

Event grouping	What events are logged	How to identify related log events	Additional information
Developer	Create Developer	<ul style="list-style-type: none"> <li>action: create</li> <li>objectType: DEVELOPER</li> </ul> <p>Audit Log</p> <p>Example:</p>	<a href="#">Onboard an Application Developer [page 663]</a>

#### Output Code

```
{
 "uuid": "<msg
GUID>",
 "user": "<user e-mail>",
 "time": "2021-06-09T10:34:32.369Z",
 "id": "<ID>"
 "success": true,
 "object": {
 "type": "DEVELOPER",
 "id": "<id>",
 "class": "com.sap.apimgmt.asmprov.core.processor.DeveloperProcessor",
 "objectId": "DEVELOPER",
 "action": "create",
 "message": "<based on the event the corresponding message will be logged>"
 }
}
```

Event grouping	What events are logged	How to identify related log events	Additional information
Developer	Update Developer	<ul style="list-style-type: none"> <li>• action: update</li> <li>• objectType: DEVELOPER</li> </ul> <p>Audit Log:</p> <p>Example:</p>	

↔ Output Code

```

{"\ "uuid\ ": "\ <
msg
GUID> \", \"user
\ ": \"<user e-
mail> \",
\"time\ ": \"202
1-06-09T10:34:
32.369Z \",
\"id\ ": \"<ID> \
\"
\"success\ ": tr
ue, \"object\ ":
{ \"type\ ": \"PR
OXY \",
\"id\ ":
{ \"class\ ": \"c
om.sap.apimgmt
.asmprov.core.
processor.Deve
loperProcessor
\",
\"objectId\ ": \
\"APPLICATION\
\", \"action\ ": \
update \",
\"message\ ": \
<based on the
event the
corresponding
message will
be logged> }"

```

Event grouping	What events are logged	How to identify related log events	Additional information
Developer	Delete Developer	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: DEVELOPER</li> </ul> <p>Audit Log:</p> <p>Example:</p>	

↔ Output Code

```
{
 \"uuid\": \"<msg
 GUID>\", \"user
 \": \"<user e-
 mail>\",
 \"time\": \"202
 1-06-09T10:34:
 32.369Z\",
 \"id\": \"<ID>\"
 \"success\": tr
 ue, \"object\":
 {\"type\": \"DE
 VELOPER\",
 \"id\":
 {\"class\": \"c
 om.sap.apimgmt
 .asmprov.core.
 processor.Deve
 looperProcessor
 \",
 \"objectId\": \"
 DEVELOPER\", \"
 action\": \"de
 lete\",
 \"message\": \"
 <based on the
 event the
 corresponding
 message will
 be logged>\"
}
```



Event grouping	What events are logged	How to identify related log events	Additional information
Environment	Create Environment	<ul style="list-style-type: none"> <li>• action: create</li> <li>• objectType: ENVIRONMENT</li> </ul> <p>Audit Log:</p> <p>Example:</p>	

↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>
"
"success":tr
ue,"object":
{"type":"EN
VIRONMENT",
"id":
{"class":"c
om.sap.apimgmt
.asmprov.model
.listener.Envi
ronmentEntityL
istener",
"objectId":"
ENVIRONMENT",
"action":"
delete",
"message":"
<based on the
event the
corresponding
meassage will
be logged>}"
```

Event grouping	What events are logged	How to identify related log events	Additional information
Environment	Delete Environment	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: ENVIRONMENT</li> </ul> <p>Audit Log:</p> <p>Example:</p>	

↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>
"
"success":tr
ue,"object":
{"type":"EN
VIRONMENT",
"id":
{"class":"c
om.sap.apimgmt
.asmprov.model
.listener.Envi
ronmentEntityL
istener",
"objectId":"
ENVIRONMENT",
"action":"
delete",
"message":"
<based on the
event the
corresponding
meassage will
be logged>}"
```

Event grouping	What events are logged	How to identify related log events	Additional information
XProperty	Create XProperty	<ul style="list-style-type: none"> <li>• action: create</li> <li>• objectType: XPROPERTY</li> </ul> <p>Audit Log:</p> <p>Example:</p>	

↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
\"success\":tr
ue,\"object\":
{\"type\":\"XP
ROPERTY\",
\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.model
.listener.XPro
pertyEntityLis
tener\",
\"objectId\":\
\"XPROPERTY\",
\"action\":\
\"create\",
\"message\":\
<based on the
event the
corresponding
message will
be logged>}"
```

Event grouping	What events are logged	How to identify related log events	Additional information
XProperty	Delete XProperty	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: XPROPERTY</li> </ul> <p>Audit Log:</p> <p>Example:</p>	

#### ↔ Output Code

```

{"\ "uuid\ ": "\ <
msg
GUID> \", "\ user
\ ": "\ <user e-
mail> \",
\ "time\ ": "\ 202
1-06-09T10:34:
32.369Z \",
\ "id\ ": "\ <ID> \
"
\ "success\ ": tr
ue, \ "object\ ":
{ \ "type\ ": \ "XP
ROPERTY \",
\ "id\ ":
{ \ "class\ ": \ "c
om.sap.apimgmt
.asmprov.model
.listener.XPro
pertyEntityLis
tener \",
\ "objectId\ ": \
"XPROPERTY \", \
\ "action\ ": \ "de
lete \",
\ "message\ ": \ "
<based on the
event the
corresponding
message will
be logged> }"

```

Event grouping	What events are logged	How to identify related log events	Additional information
XTenantProperty	Create XTenantProperty	<ul style="list-style-type: none"> <li>• action: create</li> <li>• objectType: XTENANT-PROPERTY</li> </ul> <p>Audit Log:</p> <p>Example:</p>	

#### ↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>
"
"success":tr
ue,"object":
{"type":"XP
ROPERTY","
"id":
{"class":"c
om.sap.apimgmt
.asmprov.model
.listener.XTen
antPropertyEnt
ityListener","
"objectId":"
XTENANTPROPER
TY","action\
":"create","
"message":"
<based on the
event the
corresponding
meassage will
be logged>}"
```

Event grouping	What events are logged	How to identify related log events	Additional information
XTenantProperty	Delete XTenantProperty	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: XTENANT-PROPERTY</li> </ul> <p>Audit Log:</p> <p>Example:</p>	

↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
\"success\":tr
ue,\"object\":
{\"type\":\"XT
ENANTPROPERTY\"
\",
\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.model
.listener.XTen
antPropertyEnt
ityListener\"
,\"objectId\":
\"XTENANTPROPER
TY\"},\"action\"
: \"delete\",
\"message\": \"
<based on the
event the
corresponding
message will
be logged>\"}
```

Event grouping	What events are logged	How to identify related log events	Additional information
Application	Create Application	<ul style="list-style-type: none"> <li>• action: create</li> <li>• objectType: APPLICATION</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```
"message":
{"\\"uuid\\"":"\<
msg
GUID>","\\"user
\\"":"\<user e-
mail>","\\"
\\"time\\"":"202
1-07-13T05:58:
25.864Z","\\"
\\"id\\"":"\<msg
GUID>","\\"
\\"object\\"":
{"\\"type\\"":"AP
PLICATION","\\"
id\\"":
{"\\"class\\"":"c
om.sap.it.spc.
apingmt.entity
handlers.Domai
nHook","\\"
\\"objectId\\"":"
"Application"
","\\"action\\"":"
create","\\"
\\"message\\"":"
application.up
date","\\"user\
":"\<user e-
mail>"}},
\\"attributes\\"
:
[{"\\"Applicatio
n
Id\\"":"\\"value\
"}],\\"status\\"
:"BEGIN",
\\"category\\"":"
"audit.configu
ration","\\"ten
ant\\"":"tenant
id","\\"customD
etails\\"":{}}
```

Event grouping	What events are logged	How to identify related log events	Additional information
Application ID	Update Application ID	<ul style="list-style-type: none"> <li>• action: update</li> <li>• objectType: APPLICATION</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```
"message":
{"\\"uuid\\"":"\<
msg
GUID>","\\"user
\\"":"\<user e-
mail>","\\"
\\"time\\"":"202
1-07-13T05:58:
25.864Z","\\"
\\"id\\"":"\<msg
GUID>","\\"
\\"object\\"":
{"\\"type\\"":"AP
PLICATION","\\"
id\\"":
{"\\"class\\"":"c
om.sap.it.spc.
apingmt.entity
handlers.Domai
nHook","\\"
\\"objectId\\"":"
"Application"
","\\"action\\"":"
update","\\"
\\"message\\"":"
application.up
date","\\"user\
":"\<user e-
mail>"}},
\\"attributes\\"
:
[{"\\"Applicatio
n
Id\\"":"\\"value\
"}],\\"status\\"
:"BEGIN",
\\"category\\"":"
"audit.configu
ration","\\"ten
ant\\"":"tenant
id","\\"customD
etails\\"":{}}
```



Event grouping	What events are logged	How to identify related log events	Additional information
Application ID	Delete Application ID	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: APPLICATION</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```
"message":
"{\"uuid\": \"<msg
GUID>\", \"user
\": \"<user e-
mail>\",
\"time\": \"202
1-07-13T05:58:
25.864Z\", \"id
\": \"<msg
GUID>\",
\"object\":
{ \"type\": \"AP
PLICATION\", \"
id\":
{ \"class\": \"c
om.sap.it.spc.
apingmt.entity
handlers.Domai
nHook\",
\"objectId\": \"
Application\",
\"action\": \"
delete\",
\"message\": \"
application.de
lete\", \"user\
\": \"<user e-
mail>\"}},
\"attributes\"
:
[{ \"Applicatio
n
Id\": \"value\
\"}], \"status\"
: \"BEGIN\",
\"category\": \"
audit.configu
ration\", \"ten
ant\": \"tenant
id\", \"customD
etails\": {} }
```

Event grouping	What events are logged	How to identify related log events	Additional information
Subscription	Create Subscription	<ul style="list-style-type: none"> <li>• action: create</li> <li>• objectType: SUBSCRIPTION</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```
"message":
{"\\"uuid\\"":"\<
msg
GUID>","\\"user
\\"":"\<user e-
mail>","\\"
\\"time\\"":"202
1-07-13T05:58:
25.864Z","\\"id
\\"":"\<msg
GUID>","\\"
\\"object\\"":
{"\\"type\\"":"us
er","\\"id\\"":
{"\\"class\\"":"c
om.sap.it.spc.
apingmt.entity
handlers.Domai
nHook","\\"
\\"objectId\\"":\
"Subscription\
","\\"action\\"":\
"aboutToCreate
","\\"
\\"message\\"":\
"subscription.c
reate","\\"user
\\"":"\<user e-
mail>"}}},
\\"attributes\\"
:
[{"\\"Applicatio
n
Id\\"":"\\"value\
"}],\\"status\\"
:"BEGIN","\\"
\\"category\\"":\
"audit.configu
ration","\\"ten
ant\\"":"tenant
id","\\"customD
etails\\"":{}}
```

Event grouping	What events are logged	How to identify related log events	Additional information
Subscription	Update Subscription	<ul style="list-style-type: none"> <li>• action: update</li> <li>• objectType: SUBSCRIPTION</li> </ul> <p>Audit Log</p> <p>Example:</p>	

```

↔ Output Code

"message":
{"\\"uuid\\"":\\"<
msg
GUID>\\",\\"user
\\":\\"<user e-
mail>\\",
\\"time\\"":\\"202
1-07-13T05:58:
25.864Z\\",\\"id
\\":\\"<msg
GUID>\\",
\\"object\\":
{"\\"type\\"":\\"us
er\\",\\"id\\"":
{"\\"class\\"":\\"c
om.sap.it.spc.
apingmt.entity
handlers.Domai
nHook\\",
\\"objectId\\"":\\"
Subscription\\
\\",\\"action\\"":\\"
aboutToUpdate
\\",
\\"message\\"":\\"
application.up
date\\",\\"user\\
\\":\\"<user e-
mail>\\"}},
\\"attributes\\"
:
[{"\\"Applicatio
n
Id\\"":\\"value\\
\\",\\"isSubscrib
ed\\"":\\"false\\"
)}],\\"status\\"
:\\\"BEGIN\\",
\\"category\\"":\\"
audit.configu
ration\\",\\"ten
ant\\"":\\"tenant
id\\",\\"customD
etails\\"":{}}

```

Event grouping	What events are logged	How to identify related log events	Additional information
Key Map Entry	Create Key Map Entry	<ul style="list-style-type: none"> <li>• action: create</li> <li>• objectType: KEY MAP ENTRY</li> </ul> <p>Audit Log</p> <p>Example:</p>	<a href="#">Key Value Map [page 554]</a>

↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
\",
\"success\":tr
ue,\"object\":
{\"type\":\"KE
Y MAP ENTRY\"
,\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.core.
processor.KeyM
apEntryProcess
or\"
,\"action\":\"c
reate\"
,\"message\":\"
<based on the
event the
corresponding
meassage will
be logged>\"
```

Event grouping	What events are logged	How to identify related log events	Additional information
Key Map Entry	Update Key Map Entry	<ul style="list-style-type: none"> <li>• action: update</li> <li>• objectType: KEY MAP ENTRY</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
\",
\"success\":tr
ue,\"object\":
{\"type\":\"KE
Y MAP ENTRY\"
,\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.core.
processor.KeyM
apEntryProcess
or\"
,\"objectId\":
\"KeyMapEntry\"
,\"action\":
update\"
,\"message\":
<based on the
event the
corresponding
meassage will
be logged>}"
```

Event grouping	What events are logged	How to identify related log events	Additional information
Key Map Entry	Delete Key Map Entry	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: KEY MAP ENTRY</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
\",
\"success\":tr
ue,\"object\":
{\"type\":\"KE
Y MAP ENTRY\"
,\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.core.
processor.KeyM
apEntryProcess
or\"
,\"objectId\":
\"KeyMapEntry\"
,\"action\":
delete\"
,\"message\":
<based on the
event the
corresponding
meassage will
be logged>\"}
```

Event grouping	What events are logged	How to identify related log events	Additional information
Key Map Value	Create Key Map Value	<ul style="list-style-type: none"> <li>• action: create</li> <li>• objectType: KEY MAP VALUE</li> </ul> <p>Audit Log</p> <p>Example:</p>	<a href="#">Create a Key Value Map [page 555]</a>

#### ↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
\",
\"success\":tr
ue,\"object\":
{\"type\":\"KE
Y MAP VALUE\"
,\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.core.
processor.KeyM
apValueProcess
or\"
,\"action\":\"c
reate\"
,\"message\":\"
<based on the
event the
corresponding
meassage will
be logged>\"
```

Event grouping	What events are logged	How to identify related log events	Additional information
Key Map Value	Update Key Map Value	<ul style="list-style-type: none"> <li>action: update</li> <li>objectType: KEY MAP VALUE</li> </ul> <p>Audit Log</p> <p>Example:</p>	<a href="#">Update a Key Value Map [page 556]</a>

↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
\",
\"success\":tr
ue,\"object\":
{\"type\":\"KE
Y MAP VALUE\"
,\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.core.
processor.KeyM
apValueProcess
or\"
,\"objectId\":
\"KeyMapEntry\"
,\"action\":
update\"
,\"message\":
<based on the
event the
corresponding
meassage will
be logged>\"}
```



Event grouping	What events are logged	How to identify related log events	Additional information
Key Map Value	Delete Key Map Value	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: KEY MAP VALUE</li> </ul> <p>Audit Log</p> <p>Example:</p>	<a href="#">Delete a Key Value Map [page 557]</a>

#### ↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
\",
\"success\":tr
ue,\"object\":
{\"type\":\"KE
Y MAP VALUE\"
,\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.core.
processor.KeyM
apValueProcess
or\"
,\"objectId\":
\"KeyMapEntry\"
,\"action\":
delete\"
,\"message\":
<based on the
event the
corresponding
meassage will
be logged>\"}
```

Event grouping	What events are logged	How to identify related log events	Additional information
Certificate Store	Create Certificate Store	<ul style="list-style-type: none"> <li>• action: create</li> <li>• objectType: CERTIFICATE STORE</li> </ul> <p>Audit Log</p> <p>Example:</p>	<div data-bbox="826 622 1037 658" style="border: 1px solid #ccc; padding: 5px;"> <p>↔ Output Code</p> <pre data-bbox="842 689 1053 1456"> {"\uid\":"&lt;msg GUID&gt;",\user \":"&lt;user e- mail&gt;", "time\":"202 1-06-09T10:34: 32.369Z", "id\":"&lt;ID&gt;" , "success\":tr ue,\object\": {"type\":"CE RTIFICATE STORE", "id\": {"class\":"c om.sap.apimgmt .asmprov.core. processor.Cert ificateStorePr ocessor", "action\":"c reate", "message\":" &lt;based on the event the corresponding meassage will be logged&gt;}" </pre> </div>

Event grouping	What events are logged	How to identify related log events	Additional information
Certificate Store	Update Certificate Store	<ul style="list-style-type: none"> <li>• action: update</li> <li>• objectType: CERTIFICATE STORE</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
\",
\"success\":tr
ue,\"object\":
{\"type\":\"CE
RTIFICATE
STORE\",
\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.core.
processor.Cert
ificateStorePr
ocessor\",
\"objectId\":\
\"CertificateSt
ore\", \"action
\":\"update\",
\"message\":\
<based on the
event the
corresponding
message will
be logged>\"
```

Event grouping	What events are logged	How to identify related log events	Additional information
Certificate Store	Delete Certificate Store	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: CERTIFICATE STORE</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```
"{"uuid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z",
"id":"<ID>\"
\",
\"success\":tr
ue,\"object\":
{\"type\":\"CE
RTIFICATE
STORE\",
\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.core.
processor.Cert
ificateStorePr
ocessor\",
\"objectId\":\
\"CertificateSt
ore\", \"action
\":\"delete\",
\"message\":\
<based on the
event the
corresponding
message will
be logged>\"
```

Event grouping	What events are logged	How to identify related log events	Additional information
Certificate	Create Certificate	<ul style="list-style-type: none"> <li>• action: create</li> <li>• objectType: CERTIFICATE</li> </ul> <p>Audit Log</p> <p>Example:</p>	<a href="#">Manage Certificates [page 558]</a>

↔ Output Code

↔ Output Code

```
"{"uuid\":"
 \

```

Event grouping	What events are logged	How to identify related log events	Additional information
Certificate	Update Certificate	<ul style="list-style-type: none"> <li>• action: update</li> <li>• objectType: CERTIFICATE</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

↔ Output Code

```

{"\ "uuid\ ":
 \ "<msg
 GUID>\ ", \ "u
 ser\ ": \ "<us
 er e-
 mail>\ ",
 \ "time\ ": \ "
 2021-06-09T
 10:34:32.36
 9Z\ ",
 \ "id\ ": \ "<I
 D>\ ",
 \ "success\ "
 : true, \ "obj
 ect\ ":
 { \ "type\ ": \
 "CERTIFICAT
 E\ ",
 \ "id\ ":
 { \ "class\ ":
 \ "com.sap.a
 pimgmt.asmp
 rov.core.pr
 ocessor.Cer
 tificatePro
 cessor\ ",
 \ "objectId\
 ": \ "Certifi
 cate\ ", \ "ac
 tion\ ": \ "up
 date\ ",
 \ "message\ "
 : \ "<based
 on the
 event the
 correspondi
 ng
 message
 will be
 logged>}"

```

Event grouping	What events are logged	How to identify related log events	Additional information
Certificate	Delete Certificate	<ul style="list-style-type: none"> <li>• action: delete</li> <li>• objectType: CERTIFICATE</li> </ul> <p>Audit Log</p> <p>Example:</p>	

↔ Output Code

```
"{"uid":"<msg
GUID>","user
":"<user e-
mail>","
"time":"202
1-06-09T10:34:
32.369Z","
"id":"<ID>\"
,
\"success\":tr
ue,\"object\":
{\"type\":\"CE
RTIFICATE\",
\"objectId\":\
\"Certificate\"
,\"id\":
{\"class\":\"c
om.sap.apimgmt
.asmprov.core.
processor.Cert
ificateProcess
or\",
\"action\":\"d
elete\",
\"message\":\
<based on the
event the
corresponding
meassage will
be logged>}"
```

## Related Information

[Audit Logging in the Cloud Foundry Environment](#)

[Audit Logging in the Neo Environment](#)

## 1.14 Monitoring and Troubleshooting

Information on creating tickets for reporting incidents and errors.

### Reporting an Incident

You can report an incident or error through the [SAP Support Portal](#). For more information, see [Product Support](#).

Please use the following component for your incident:

Component Name	Component Description
OPU-API-OD	SAP API Management - On Demand

When submitting the incident we recommend including the following information:

- Landscape information (Canary, EU10, US10)
- The URL of the page where the incident or error occurs
- The steps or clicks used to replicate the error
- Screen grabs, videos, or the code being inputted

#### Perform Application Performance Tests

As a customer, you can perform the application performance tests of SAP API Management upon mutual agreement with SAP after subscription. To conduct these tests, you need permission. For the process of obtaining the required permission, refer to the following KBA: [3215315 - Application Performance Tests by Customer for SAP API Management](#).

### Frequently Asked Questions

For the list of answers to common questions about SAP API Management, see [Frequently Asked Questions](#).

#### 1.14.1 Limits in API Management

This topic describes the product configuration and the naming conventions for API Management.

Consider the boundary conditions mentioned in the following tables for building, managing, and reviewing the APIs. API Management is designed to perform at its maximum, when it is configured within the specified conditions. Exceeding these values leads to:

- High API latency
- Low API throughput
- Failing API calls.



Currently, certain configurations are automatically enforced for optimal performance.

## Product Configurations

Feature	Specified Configuration Values	Automatically Enforced
<b>API Proxies</b>		
Port	Virtual host URL can only be configured only on the secured port 443 over https.	Yes
API proxy resource file size (such as XSL, JavaScript or Python).	15 MB	Yes
Maximum number of resources that can be attached to an API proxy	<p>You can attached up to 100 resources to an API proxy. However, it is recommended that you do not add more than 100 resources to an API proxy as it might lead to a timeout while updating or deploying an API proxy.</p> <p>In case you have a business requirement to attach more than 100 resources to an API proxy, please contact the SAP API Management support team by creating a ticket with component OPU-API-OD-DT. However, the team can support your request up to a maximum of 200 resources per API proxy.</p>	Yes
<b>Virtual Host</b>		
Additional virtual host in Cloud Foundry	By default, only 3 virtual hosts can be configured per tenant. In case you have a business requirement to have more than 3 virtual hosts within a tenant, please raise a support ticket through the <a href="#">SAP Support Portal</a> using the component OPU-API-OD-OPS.	No
<b>Cache and KVM</b>		
Caches	100 MB	No
Cache key size	2 KB	Yes
Cache value size	512 KB	Yes
Cache expiration	>=180 seconds, <= 30 days	No
Key Value Map (KVM) key size	2 KB	Yes
Key Value Map (KVM) value size	10 KB	No
<b>Keys, Developers, Apps, Products</b>		
API key size	2 KB	Yes
Custom attribute name size	255 characters	Yes

Feature	Specified Configuration Values	Automatically Enforced
Custom attribute value size	1024 characters	Yes
Number of custom attributes permitted	18	Yes
<b>OAuth</b>		
OAuth access token expiration	>= 180 seconds, <= 30 days	No
OAuth refresh token expiration	>= 1 day, <= 90 days	No
OAuth access and refresh token size	2 KB	Yes
<b>System</b>		
API proxy request URL size	7 KB	Yes
Request header size	18 KB	Yes
Response header size	25 KB	Yes
Request size (for non-streamed HTTP requests)	10 MB	Yes
Request size (for streamed HTTP requests)	<500 MB. However, if the connection is terminated unexpectedly, you must reinitiate the connection.	No
Response size (for non-streamed HTTP requests)	10 MB	Yes
Timeout ( Applicable for all API Proxies and the Backend Server is expected to respond within the value set )	55 Seconds	Yes
Security Protocol	TLSv 1.2 only ( on Hyperscalers )	Yes
	TLSv 1.2 & TLSv1.1 ( on SAP DC )	Yes
	TLSv1.1 (deprecated, planned to be removed by end of 2019)	No
	TLSv1.0 (Unsupported)	Yes

Feature	Specified Configuration Values	Automatically Enforced
SNI ( Server Name Indication ) Enforcement	<p>In API Management, the client is expected to pass a SNI extension (server_name) with target endpoint hostname as part of the initial TLS handshake. Based on the server_name SNI extension the APIM servers will determine the certificate that would be served to the client.</p> <p>For Client which doesn't support SNI extension, APIM would send a default certificate which is provided by SAP.</p> <p>For Reference on SNI : <a href="https://en.wikipedia.org/wiki/Server_Name_Indication">https://en.wikipedia.org/wiki/Server_Name_Indication</a></p>	Yes
Security Protocol applicable for API Management Runtime	TLSv1.2 only ( on Hyperscalers & SAP DC )	Yes
	TLSv1.0 & TLS 1.1 (Unsupported)	Yes

## Naming Conventions

Table below describes the naming constraints for API Management.

Name	Maximum Characters	Permitted Characters
API product		Alphanumeric, space, and the following: _ - . # \$ %
Cache name	255	Alphanumeric
Developer app		Alphanumeric, space, and the following: _ - . # \$ %
E-mail ID		Valid e-mail address syntax
Policy name	255	Alphanumeric and underscore.
Resource file names.	255	Alphanumeric, space, and the following: : / \ ! @ # \$ % ^ & { } [ ] ( ) _ + - = , . ~ ' "
Revision name	5	Numeric

## 1.14.2 Monitor the Health of Custom Domain Virtual Host Certificates Using SAP Cloud ALM

You can use SAP Cloud Application Lifecycle Management (ALM) application to proactively detect issues and monitor the health of custom domain virtual host certificates in API Management.

The SAP Cloud Application Lifecycle Management (ALM) platform allows you to monitor environment backlogs and status of automation processes regarding execution status, application status, start delay, and runtime of various SAP Cloud solutions and services. To integrate the Cloud Application Lifecycle Management (ALM) application with the Health service endpoint, refer [SAP Cloud ALM Onboarding](#) .

The API <https://<host>:<port>/api/1.0/Health>, which has been provisioned in API Management, can be integrated with Cloud Application Lifecycle Management (ALM) to provide information about the custom domain virtual host certificates expiry details. You can use this API to read information about the certificates provided by the customers for their custom domain virtual hosts.

### Note

You have to enable API access with *APIPortal. Administrator* role to use the <https://<host>:<port>/api/1.0/Health> API. To access this API from the Cloud Application Lifecycle Management (ALM) application, use *OAuth* authentication.

## 1.15 Migration of API Management Content

You can choose to clone the API Management content from Neo to Cloud Foundry or between different Cloud Foundry environments.

This table summarizes the migration strategy that we currently support:

Migration Strategy	Types of Migration	Use Cases	More Details
Migration of API Management content from Neo to Cloud Foundry	Standard Migration		<a href="#">Migrating API Management from Neo to Cloud Foundry Environment [page 781]</a>
	Starter Plan Migration (Runtime Reuse Design time Only Migration)	Migration of API Management content within the same Cloud Foundry subaccount	<a href="#">Migrating API Management Subscription Created Using the Starter Plan Service Instance [page 813]</a>
		Migration of API Management content between different Cloud Foundry subaccounts	<a href="#">Migrating API Management Subscription Created Using the Starter Plan Service Instance to Different Subaccounts [page 815]</a>

Migration Strategy	Types of Migration	Use Cases	More Details
Migration of API Management content from one Cloud Foundry environment to another Cloud Foundry environment	Standard Migration		<a href="#">Migration of API Management Content between Cloud Foundry Environments [page 817]</a>

## 1.15.1 Migrating API Management from Neo to Cloud Foundry Environment

You can migrate the API Management content in Neo environment to a public cloud infrastructure (hyperscalers) within the Cloud Foundry environment.

Migration Assistant for asset migration includes the tools and utilities that enable migration of design time assets nondisruptively from the Neo to the Cloud Foundry environment.

Your source system is the system that contains your API Management content in the Neo environment.

Your target system is the system that hosts your API Management content on the hyperscalers-managed infrastructure within the Cloud Foundry environment. Here Cloud Foundry environment can be your native standalone API Management subscription or API Management capability within the Integration Suite.

For the migration assistance, you must have an Integration Suite subscription with API Management capability enabled within Integration Suite.

After completing the prerequisites mentioned in the steps below, you can clone your API Management artifacts nondisruptively from the source to the target system. Post cloning, you must complete some user actions and validate your target system.

### Note

The Developer portal is renamed to API business hub enterprise in Cloud Foundry environment. In this document API business hub enterprise is referred to as Developer portal even in Cloud Foundry environment.

The steps assisting the migration of your API Management from your source system to a target system are:

1. [Prerequisites \[page 782\]](#)
2. [Clone API Management Content \[page 783\]](#)
3. [Post Cloning Tasks \[page 805\]](#)

## 1.15.1.1 Prerequisites

Checks to be completed before you start migrating your API Management content nondisruptively from your source system to a target system.

- Your source system is the system that has your API Management subscription in the Neo environment.
- Your target system is the system that has your API Management content on the hyperscalers-managed infrastructure within the Cloud Foundry environment.

### Prerequisites for the source system

- You must have a valid API Management system (API portal and Developer Portal) running in the Neo environment.
- The source system must support basic authentication for API access on and API business hub enterprise(which is the developer portal).
- Make a note of the and API business hub enterprise(developer portal) URLs of the source system and keep it handy.
- You must have identified a user with the following roles assigned in your source systems:
  - APIPortal.Administrator
  - AuthGroup.API.Admin role

Keep the credentials of this user handy. These credentials are used while filling in the details of the `apim-tct-input.json` file before running the Tenant Cloning Tool. See [Clone API Management Content \[page 783\]](#).

### Prerequisites for the target system

- If API Management is not already enabled on your target system, complete the set-up. See [Initial Setup](#) and [Enable API Management Capability](#) . Check whether the API Management service broker service instance is created with the Starter Plan in the same subaccount.

Service Instance for Starter Plan in the Subaccount	Instruction
Already present	<p>You cannot use this subaccount for migration. Create a new subaccount in the hyperscalers-managed infrastructure within the cloud foundry environment and enable API Management on that subaccount for it to act as your target system.</p> <p>Additionally, if you want to reuse the existing runtime then follow the steps mentioned in the <a href="#">Migrating API Management Subscription Created Using the Starter Plan Service Instance [page 813]</a>.</p>
Not present	<p>You can choose to reuse this account as your target system for migration, or create a new subaccount.</p>

If you have already enabled API Management on your target system, and want to reuse the same for migration:

- It's recommended that you do not have any pre-existing entities such as API proxies or products on this system.

### Note

Any entity, if pre-existing in your target API Management capability, can be over-written during the cloning process.

- If your target system is connected to a custom IDP, ensure that your IDP is configured correctly, and mapping for the details like your first name, last name, email ID, and user ID is done.

### Note

Please ensure that the application developer's attributes, like first name, last name, email ID, and user ID, are identical in both source and target identity providers. In API Management, the application developer's attributes are case-sensitive.

Consider the following example: During cloning, the email address `john.smith@abc.com` in the source becomes `John.Smith@abc.com` in target due to the change in configurations in Custom IDP. This mismatch might lead to data discrepancy during application creation and metering in the target after cloning.

- Ensure that API access is enabled for the and the API business hub enterprise(developer portal) for the following roles:
  - APIPortal.Administrator
  - AuthGroup.API.AdminMake a note of the service keys (**url**, **tokenurl**, **clientId**, and **clientSecret**) for the given roles, and keep handy. To know more about API access plans for API portal, see [Accessing API Management APIs Programmatically](#). To know more about API access plan for API business hub enterprise, see [Accessing API business hub enterprise APIs Programmatically](#), without which the cloning of the API business hub enterprise entities might fail.
- When you have API products protected by the custom roles permission in the source Neo system, ensure that custom roles creation and assignments are done in the target Cloud Foundry environment before starting the migration.

Once you complete these checks, you can start cloning your API Management content from the source to the target system. See [Clone API Management Content \[page 783\]](#).

## 1.15.1.2 Clone API Management Content

Clone the API Management content using the Tenant Cloning tool.

Once you have your source and target system ready, you can clone your API Management content to the target system by running the Tenant Cloning Tool that you downloaded from [here](#).

### Prerequisites

- You must have downloaded the Tenant Cloning Tool ( ) from the link provided above.APIM-TCT-  
<version>.zip

- APIM-TCT-You must have extracted the contents of the `APIM-TCT-<version>.zip` file into a folder (example name `apim-tct`).

This extracted folder must contain:

- a `java apim-tct-client-<version>.jar` file
- a sample `apim-tct-input.json` file
- a `lib` folder (this folder and its contents must not be modified)
- a `README.md` file
- Script files to download open-source libraries that are required to run the `apim-tct-client-<version>.jar` file:
  - `download_dependencies.ps1` for Windows systems
  - `download_dependencies.sh` for Mac and Linux systems

#### Note

Download the dependencies as described in the **Downloading the Dependencies** section.

#### Note

If you are using the version of the Tenant Cloning Tool prior to 1.5.2, make sure that you update to the latest version 1.5.2 or above. This is done to handle the critical vulnerability CVE-2021-44228 and CVE-2021-45046, which was detected in the open-source library log4j2.

- The system running the API Management Tenant Cloning Tool must have Java Runtime Environment 8 or above supported.
- Microsoft Excel File Reader

## Downloading the Dependencies

### For Windows Systems:

- Open the PowerShell terminal.
- Go to the `apim-tct` folder in the terminal.
- Run the `.\download_dependencies.ps1` command.  
The required libraries are downloaded to the `lib` folder.

### For Mac and Linux Systems:

- Open the default terminal from your system.
- Go to the `apim-tct` folder in the terminal.
- Run the `chmod +x download_dependencies.sh` command to make the file executable.
- Run the `.\download_dependencies.sh` command.  
The required libraries are downloaded to the `lib` folder.

#### Note

If you encounter an error while running these commands, then you can download the dependencies manually from the link provided in the script file and place them into the `lib` folder.



## Procedure

1. Fill in the `apim-tct-input.json`

Ensure that you don't modify the name of the `apim-tct-input.json` file.

For more information on how to create the service key, refer the [Accessing API Management APIs Programmatically \[page 127\]](#) and [Accessing API business hub enterprise APIs Programmatically \[page 134\]](#).

Structure of the `apim-tct-input.json` file:

Input Field		Credentials Type	Data Type	Required/Optional	Description
source	apiportal	<code>url</code>	String	Required	URL of the source API management, API portal in the Neo environment  Example: <code>https://&lt;application_name&gt;&lt;provider_subaccount&gt;-&lt;consumer_subaccount&gt;.&lt;domain&gt;</code>
		<code>username</code>	String	Optional	User ID having the <b>APIPortal.Administrator</b> role in the above subscription  You're prompted to enter these values while running the command in Step 3 if you have not already provided these details in the <code>apim-tct-input.json</code> file.

Input Field	Credentials Type	Data Type	Required/Optional	Description
	<code>password</code>	String	Optional	<p>Password of the above user</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>
<code>devportal</code>	<code>url</code>	String	Required	<p>URL of the source API Management, Developer Portal in the Neo environment</p> <p>Example:  <code>https://&lt;application_name&gt;&lt;provider_subaccount&gt;-&lt;consumer_subaccount&gt;.&lt;domain&gt;</code></p>

Input Field	Credentials Type	Data Type	Required/Optional	Description
<code>username</code>	Basic	String	Optional	User ID having the <code>AuthGroup.API.Admin</code> role in the above subscription  You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.
<code>password</code>		String	Optional	Password of the above user  You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.

Input Field	Credentials Type	Data Type	Required/Optional	Description
cfSubaccount-TenantID	Supported values: "guid"	String	Optional	This is the Tenant ID for your Cloud Foundry sub account where starter plan service instance is enabled.


#### Note

If you are migrating within the same sub-account, you are not required to add this parameter.

This parameter is mandatory if you are migrating to a Cloud Foundry sub-account, which is different from your existing starter plan sub-account.

#### Note

Navigate to the cockpit to fetch the Cloud Foundry Tenant ID

Input Field	Credentials Type	Data Type	Required/Optional	Description		
				for the subaccount where the starter plan service instance exists. 		
target	apiportal	<b>Url</b>	String	Required	URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role	
		<b>tokenUrl</b>	Client Secret	String	Required	Token URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role

**Note**  
Choose the relevant fields based on the credential type you've configured for the API access plan. For example, if you've used Client Secret as the credential type, do not select the fields from X509 mTLS.

Input Field	Credentials Type	Data Type	Required/Optional	Description
<code>clientId</code>		String	Optional	<p>The client ID received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>
<code>clientSecret</code>		String	Optional	<p>The client secret received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>

Input Field	Credentials Type	Data Type	Required/Optional	Description
<code>certurl</code>	X509 mTLS	String	Optional	Cert URL received during creation of the service key for API portal API access for the APIPortal.Administrator role.
<code>certificate</code>		String	Optional	The content of the certificate received during creation of the service key for API portal API access for the APIPortal.Administrator role.
<code>clientid</code>		String	Optional	Client ID received during creation of the service key for API portal API access for the APIPortal.Administrator role.
<code>privatekey</code>		String	Optional	Private Key received during creation of the service key for API portal API access for the APIPortal.Administrator role.

Input Field		Credentials Type	Data Type	Required/Optional	Description
apiportalSelf-ServiceAdmin	<b>Url</b>	Client Secret	String	Required	URL received during creation of the service key for API portal API access for the <b>APIManagement.SelfService.Administrator</b> role.
<p><b>Note</b></p> <p>Choose the relevant fields based on the credential type you've configured for the API access plan. For example, if you've used Client Secret as the credential type, do not select the fields from X509 mTLS.</p>	<b>tokenUrl</b>		String	Required	Token URL received during creation of the service key for API portal API access for the <b>APIManagement.SelfService.Administrator</b> role.
	<b>clientId</b>		String	Optional	The client ID received during creation of the service key for API portal API access for the <b>APIManagement.SelfService.Administrator</b> role.



Input Field	Credentials Type	Data Type	Required/Optional	Description
<code>clientSecret</code>		String	Optional	<p>The client secret received during creation of the service key for API portal API access for the <b>APIManagement.SelfService.Administrator</b> role.</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>
<code>certurl</code>	X509 mTLS	String	Optional	<p>Cert URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role.</p>
<code>certificate</code>		String	Optional	<p>The content of the certificate received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role.</p>

Input Field		Credentials Type	Data Type	Required/Optional	Description
	<code>clientid</code>		String	Optional	Client ID received during creation of the service key for API portal API access for the APIPortal.Administrator role.
	<code>privatekey</code>		String	Optional	Private Key received during creation of the service key for API portal API access for the APIPortal.Administrator role.
devportal	<code>url</code>	Client Secret	String	Required	URL received during creation of the service key for Developer Portal API access for the <b>AuthGroup.API.Admin</b> role.
	<code>tokenUrl</code>		String	Required	Token url received during creation of the service key for API business hub enterprise API access for the <b>AuthGroup.API.Admin</b> role.

**Note**

Choose the relevant fields based on the credential type you've configured for the API access plan. For example, if you've used Client Secret as the credential type, do not select the fields from X509 mTLS.

Input Field	Credentials Type	Data Type	Required/Optional	Description
<code>clientId</code>		String	Optional	<p>The client ID received during creation of the service key for Developer Portal API access for the <b>AuthGroup.API.Admin</b> role.</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>
<code>clientSecret</code>		String	Optional	<p>The client secret received during creation of the service key for Developer Portal API access for the <b>AuthGroup.API.Admin</b> role.</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>

Input Field	Credentials Type	Data Type	Required/Optional	Description
<code>certurl</code>	X509 mTLS	String	Optional	Cert URL received during creation of the service key for API portal API access for the APIPortal.Administrator role.
<code>certificate</code>		String	Optional	The content of the certificate received during creation of the service key for API portal API access for the APIPortal.Administrator role.
<code>clientid</code>		String	Optional	Client ID received during creation of the service key for API portal API access for the APIPortal.Administrator role.
<code>privatekey</code>		String	Optional	Private Key received during creation of the service key for API portal API access for the APIPortal.Administrator role.

Input Field	Credentials Type	Data Type	Required/Optional	Description
skipApplicationKeySecretCloning	Supported values: <b>true</b> / <b>false</b>	Boolean	Optional	<ul style="list-style-type: none"> <li>The default value for skipApplicationKeySecretCloning is false.</li> </ul>

**Note**

If you want to skip the cloning of Application Key and Secret in side by side migration, then set the `"skipApplicationKeySecretCloning"` flag to true.

Input Field	Credentials Type	Data Type	Required/Optional	Description
targetDestinationRefreshOnSwitchOver	Supported values: <b>true/</b> <b>false</b>	Boolean	Optional	The default value for targetDestinationRefreshOnSwitchOver is false.

#### Note

Add this parameter to ensure that during the switch-over stage the Tenant Cloning Tool waits for five minutes for the design time to connect to the runtime on the target API portal. If this parameter is not configured, the Tenant Cloning Tool will continue to execute the switch-over without any delay.

Input Field	Credentials Type	Data Type	Required/Optional	Description	
clone	skip-apiportal	Supported values: <b>true/</b> <b>false</b>	Boolean	Optional	<ul style="list-style-type: none"> <li>The default value for skip-apiportal is false, and API portal entities are cloned</li> <li>If you set the value for skip-apiportal to true, no cloning of the API portal entities takes place.</li> </ul>
	skip-devportal	Supported values: <b>true/</b> <b>false</b>	Boolean	Optional	<ul style="list-style-type: none"> <li>The default value for skip-devportal is false, and Developer Portal entities are cloned.</li> <li>If you set the value for skip-devportal to true, no cloning of the Developer Portal entities takes place.</li> </ul>
stage		Supported values: <b>"DEFAULT"</b>   <b>"SWITCHOVER"</b> <b>R</b>	string	Optional	The supported values for this parameter is either default or switchover.

\*\*\* apiportalSelfServiceAdmin This input field is mandatory for Starter Plan migration.

\*\*\* API portal credentials for source and target for all scenarios are mandatory.

## → Remember

For the clone input attribute:

- Both skip-apiportal and skip-devportal are set to false by default, so, API portal entities are cloned first, followed by Developer Portal entities.
- If both skip-apiportal and skip-devportal are set to true, no cloning takes place.
- If skip-apiportal is set to false, but skip-devportal is set to true, then only the API portal entities are cloned.
- If skip-apiportal is set to true, but skip-devportal to false, then only Developer Portal entities are cloned and cloning for entities (like applications) may fail, pertaining to nonavailability of dependent entity (like API Product) in Developer Portal.

Sample configuration:

```
{
 "source": {
 "apiportal": {
 "url": "<URL of Source (Neo based) API Portal>",
 "username": "<user id having APIPortal.Administrator role in
above subscription>",
 "password": "<password of the above user>"
 },
 "devportal": {
 "url": "<URL of Source (Neo based) Developer Portal>",
 "username": "<user id having AuthGroup.API.Admin role in above
subscription>",
 "password": "<password of the above user>"
 },
 "cfSubaccountTenantID": "1d1b3316-cf22-44b5-973f-d2d8a132444a"
 },
 "target": {
 "apiportal": {
 "url": "<url received during service key creation for API
Portal's API Access for APIPortal.Administrator role>",
 "tokenUrl": "<token url received during service key creation for
API Portal's API Access for APIPortal.Administrator role>",
 "clientId": "<clientId received during service key creation for
API Portal's API Access for APIPortal.Administrator role>",
 "clientSecret": "<clientSecret received during service key
creation for API Portal's API Access for APIPortal.Administrator role>"
 },
 "apiportalSelfServiceAdmin": {
 "url": "<url received during service key creation for API
Portal's API Access for APIManagement.SelfService.Administrator role>",
 "tokenUrl": "<token url received during service key creation for
API Portal's API Access for APIManagement.SelfService.Administrator role>",
 "clientId": "<clientId received during service key creation for
API Portal's API Access for APIManagement.SelfService.Administrator role>",
 "clientSecret": "<clientSecret received during
service key creation for API Portal's API Access for
APIManagement.SelfService.Administrator role>"
 },
 "devportal": {
 "url": "<url received during service key creation for Developer
Portal's API Access for AuthGroup.API.Admin role>",
 "tokenUrl": "<token url received during service key creation for
Developer Portal's API Access for AuthGroup.API.Admin role>",
 "clientId": "<clientId received during service key creation for
Developer Portal's API Access for AuthGroup.API.Admin role>",

```



```

 "clientSecret": "<clientSecret received during service key
creation for Developer Portal's API Access for AuthGroup.API.Admin role>"
 },
 "skipApplicationKeySecretCloning" : <false|true>,
 "clone": {
 "skip-apiportal": <false|true> ,
 "skip-devportal": <false|true>
 },
 "stage": <"DEFAULT" | "SWITCHOVER">
}

```

2. Run the following commands from your Java command-line interface to verify the setup and check the version of the tool. This is an optional step.

- To verify the setup:  
`java -jar apim-tct-client-<version>.jar verify`
- To check the version of the tenant cloning tool you're using:  
`java -jar apim-tct-client-<version>.jar version`

3. To begin the cloning process, run the following command from your Java command-line interface:

```
java -jar apim-tct-client-<version>.jar
```

#### Result

Your API Management entities are now cloned to your target system.

An excel file named `apimtct-output.xlsx` and a log file named `apimtct-logs.log` are generated in the same folder where the `.jar` file is present.

The status of each cloned entity is stored in a separate worksheet within the output excel file.

Structure of a Worksheet Within `apimtct-output.xlsx` File

Column	Description
ID	Entity ID
Name	Entity name
Type	Entity type
Script Execution Timestamp (UTC)	Script execution time in UTC
Artifact's Last Modified Timestamp (UTC)	Last modified time of the entity in the source API Management system (UTC)
STATUS	Migration Status: <ul style="list-style-type: none"> <li>• SUCCESS (Entity successfully cloned)</li> <li>• FAILURE (Entity failed to clone)</li> <li>• SKIPPED (Cloning of Entity skipped)</li> </ul>

You can view the status of the cloned content in the `apimtct-output.xlsx` file or in the `apimtct-logs.log` file.

#### Note

- Ensure that the `apimtct-output.xlsx` file isn't open while you run the script.
- It's recommended that you don't modify the `apimtct-output.xlsx` file.

#### Troubleshooting During Cloning:

- If the Tenant Cloning Tool shuts down unexpectedly, restart and try again.

If the tool throws an error repeatedly while running, you can report the incident or error on the component OPU-API-OD-DT through the [SAP Support Portal](#).

## Next Steps

After the cloning process completes, you must perform the tasks mentioned in the **User Actions** worksheet within the output excel file `apimtcct-output.xlsx`.

To know more about what actions you must take, see the **User Actions** section in [Post Cloning Tasks \[page 805\]](#).

To know more about the entities that are cloned and the entities that aren't cloned, see [Cloned and Uncloned Entities \[page 802\]](#).

### 1.15.1.2.1 Cloned and Uncloned Entities

Refer this section for the entities that are cloned and entities that aren't cloned during the migration process.

#### Entities That Are Cloned

##### Note

Currently, when a custom role is assigned to a Product, the Application creation using the tenant cloning tool is not supported.

As a work-around, before initiating the cloning process, remove the custom role assigned to the Product in the Source system and proceed with the cloning process.

After the cloning process is completed, reassign the custom roles to the Product in the Source system. Also, ensure that the custom roles are assigned to the Product in the Target system.

In case the custom roles aren't appearing in the [Permission](#) tab, as mentioned in the Prerequisite section, ensure that the custom roles are created and assigned to the developers in the target Cloud Foundry environment.

##### Note

If you have made any customizations to the `HelloWorld` sample proxy, and you want to migrate this proxy to the target, while cloning you might get the following error: "Unable to import API Proxy from zip file; xml content invalid" To address this, execute the following steps:

1. Export the `HelloWorld` API.
2. Open the zip file and edit the `metadata.xml` file to add the `created_by` field as shown below:

### Sample Code

```
<life_cycle>
 <changed_by>yourUserId</changed_by>
 <created_by>yourUserID</created_by>

</life_cycle>
```

Please note that your `userId` is as per your Identity Service configuration. You can find your `userId` when you open any proxies in the API portal.

3. Save the zip file.
4. Delete the existing `HelloWorld` proxy from the API portal.
5. Import this edited zip file.

With this the `created_by` will reflect in the API proxy.

The following list displays the API Management entities that are cloned:

- Certificates and Certificate Store
- Rate Plans
- Key Value Maps
- API Providers
- Policy Templates
- API Proxies
- API Products
- Measure Codes for Custom Measures
- Dimension Codes for Custom Dimensions
- Application
- Application Developer
- Access Control Permissions for API Product
- Custom Metrics and Charts

## Entities That Are Not Cloned

The following list displays the API Management entities that aren't cloned, including sensitive data like your certificates and credentials.

- **Sensitive Data**
  - Certificates
  - Encrypted Key Value Maps
  - API Provider Passwords
  - Monetization Bills

To know about the actions that you must perform for the uncloned certificates, encrypted key value maps, and API provider passwords, see the **User Actions** section in [Post Cloning Tasks \[page 805\]](#).

- **Runtime data**
  - Quota Counters

- OAuth Tokens for API Proxy runtime calls
- Runtime states of any API Management entity
- **Configurations**
  - Cloud Connector Setup
  - Custom Role creation and its assignments
  - Default role assignment to users
  - Principal Propagation setup for OpProxy
  - Any configurations created at the subaccount level
  - Any integrations with other systems (like SAP Web IDE)
  - Custom IDP Setup (if any)
  - Existing Route Bindings (if any)

To know about the actions that you must perform for these uncloned entities, see the **Actions required on Configurations** section in [Post Cloning Tasks \[page 805\]](#).

### 1.15.1.2.2 Tenant Cloning Tool Behavior

This topic describes the behavior of the Tenant Cloning Tool with respect to cloning some of the entities from your source system.

- If you add or modify an entity in your source system, it is always cloned to the target system in your subsequent run of the Tenant Cloning Tool.
- If you add a new entity to your target system at any point, it is retained in the target system after the subsequent run of the Tenant Cloning Tool, irrespective of whether the entity is present in your source system or not.
- Newer state of an existing entity present in your source system is always migrated to the target system after the subsequent run of the Tenant Cloning Tool, and overwrites any older state of the entity in the target.
- During the cloning of the Developer Portal entity **Application Developer**, the app developer receives email notifications while being onboarded to the target Developer Portal. We recommend that you inform your developers about the impending migration and email notifications that they might receive during the process.
- Custom Charts are cloned to the target as many times as you run the Tenant Cloning Tool.
- All the API proxies are cloned onto the default virtual host.
- Post cloning, the API proxies on the target system are in active and deployed state. You must reapply the desired states to the proxies.

To know more about API proxy states, see [API Proxy States \[page 580\]](#).

#### Note

If the Tenant Cloning Tool is used to clone an API proxy or a product with more than 100 resources attached to it, you might notice data inconsistency in the target system (API business hub enterprise or ). It is recommended that you do not add more than 100 resources per proxy or product. For more information, see [Limits in API Management \[page 776\]](#).

- Cloning of custom chart is now supported for migrating API Management content created using the Starter Plan service instance.

### Note

Known constraints of migrating applications:

#### Use Case 1:

You are migrating application A1, subscribed to a product P1 with no rate plans. You associate product P1 with a rate plan later in the source.

Now if you attempt to migrate application A1 from the source developer portal to the target developer portal, the application created in the target fails with an error.

**Reason for the error:** During migration, the product gets cloned along with the newly added rate plan in the target tenant. However, while creating the application in the target tenant, the system couldn't locate the rate plan ID. The rate plan ID was not present in the payload as it belonged to an older subscription, resulting in application creation failure.

#### Use Case 2:

You already have a rate plan attached to a product P1 while creating application A1. However, you decide to add a revised rate plan to product P1 after application A1 gets created. In such a scenario, application A1 will continue to have a rate plan that was present during the creation of the application. If you migrate application A1 from the source developer portal to the target developer portal, application creation will fail, and the same error as Use Case 1 will get generated.

**Reason for the error:** During migration, the product gets cloned along with the newly revised rate plan in the target tenant. However, while creating the application in the target tenant, the system couldn't locate the revised rate plan ID. The revised rate plan ID was not present in the payload as it belonged to an older subscription, resulting in application creation failure.

### Note

CacheResources cloning is currently not supported via the Tenant Cloning Tool. You must create it manually on the target using the following service: `<apiportal-host>/apiportal/api/1.0/Management.svc/CacheResources`.

## 1.15.1.3 Post Cloning Tasks

Post the completion of the cloning process, you must perform some actions, checks, and validations.

The following sections outline the tasks that need to be completed after the cloning of your API Management content from Neo to the Cloud Foundry environment.

### User Actions

You can view the status of the cloned artifacts in the `apim-tct-output.xlsx` excel file or in the `apim-tct.log` file, generated in the same folder where the `.jar` file is present.

Perform the tasks mentioned in the **User Actions** worksheet within the `apim-tct-output.xlsx` excel file.

The following table describes the actions required for each cloned entity:

Cloned Entity	User Action
Certificates	<p>All the certificates that are cloned to the target system are dummy certificates.</p> <p>Perform the following steps:</p> <ol style="list-style-type: none"><li>1. From your source system, note down the certificate names and the corresponding certificate store name.</li><li>2. From your target system, delete the dummy certificates that were cloned:<ol style="list-style-type: none"><li>1. In your target API Management, API portal, navigate to <b>Configure &gt; Certificates</b>.</li><li>2. Select the cloned dummy certificate that you want to delete.</li><li>3. Click the delete icon under the <b>Actions</b> column.</li></ol></li><li>3. In your target API Management, API portal, upload the relevant certificates, providing the same names and under same certificate store as present in your source system.<ol style="list-style-type: none"><li>1. In your target API Management, API portal, navigate to <b>Configure &gt; Certificates</b>.</li><li>2. Click <b>Create</b>.</li><li>3. In the <b>Create Certificate</b> window, provide the details and upload the certificate.</li></ol></li></ol>
Key Value Maps	<p>Fill in the values for the keys of the encrypted Key Value Maps.</p> <ol style="list-style-type: none"><li>1. In your target API Management, API portal, navigate to <b>Configure &gt; Key Value Maps</b>.</li><li>2. Click on the encrypted key value map.</li><li>3. In the <b>Edit Key Value Map</b> window, provide the details and click <b>Save</b>.</li></ol>

## Cloned Entity

## User Action

---

### API Provider Credentials

Provide the Basic Auth password (if present in your source system) for an API Provider.

1. In your target API Management, API portal, navigate to [▶ Configure ▶ API Providers ▶](#).
2. Click on the desired API provider.
3. In the *View API Provider* window, click [▶ Catalog Service Settings ▶ Edit ▶](#).
4. Provide the basic auth password for the API provider and click *Save*.

Update open connector credentials:

1. In your target API Management, API portal, navigate to [▶ Configure ▶ API Providers ▶](#).
2. Click on the desired API provider.
3. In the *View API Provider* window, click [▶ Catalog Service Settings ▶ Edit ▶](#).
4. Provide the Org ID and User Secret from your corresponding Open Connector subscription.

---

### Proxy Scoped Key Value Map

Provide instance token value in the proxy scoped Key Value Map.

1. In your target API Management, API portal, navigate to [▶ Develop ▶ APIs ▶](#).
  2. Click on the desired API proxy.
  3. In the *View API* page, scroll down to *Key Value Map Associated* and choose the Key Value Map.
  4. On the *Edit Key Value Map* page, update the *Value* with the instance token value for the corresponding open connector instance.
  5. Provide the Org ID and User Secret from your corresponding Open Connector subscription.
-

## Cloned Entity

Custom Domain based Virtual Host

### Note

This is only relevant for starter plan migration.

## User Action

If you have custom domain based virtual host in the source system, then perform the following checks to verify whether the custom domain based virtual hosts are cloned on the target systems:

If the URL of virtual hosts looks different in target, which means if the sub domain of the URL is different than the source, revert in the same migration ticket asking the Operations team to set the correct URL for this virtual host.

Please ensure that you provide the following virtual host details (from the source) to the Operations team:

- Custom domain virtual host URL
- Virtual host ID

The Operations team will use these details to update the virtual host domain in the target system so that it matches with the source.

### Note

This step has to be completed before starting the Switch Over stage.

If you have multiple customain based virtual host, perform the same procedure for each virtual host.

## Actions Required on Configurations

Depending on the configurations you have on your source system, you must configure the following in your target system:

- Custom IDP Setup (if any)
- Default role assignment to users
- Custom Role creation and its assignments
- Cloud Connector setup
- Principal Propagation setup at the subaccount level
- Changes to Principal Propagation policy for on-premise connectivity
- Migration of route service bindings. For more information, see [Migrating Route Service Binding \[page 809\]](#)
- Any integrations with other systems (like SAP Web IDE)
- Any other configurations that you created for API Management at the subaccount level of your source system

To know more about the entities that are cloned and the entities that aren't cloned, see [Cloned and Uncloned Entities \[page 802\]](#).



## Note

For the on-premise APIs, the URL of the target.basepath changes while migrating from Neo to Cloud Foundry. If you've customized any of the policies, where the target.basepath is being used, then make sure that you update the content of the policy accordingly in the target Cloud Foundry system. For example, after migration the target.basepath URL in Cloud Foundry might have an additional segment. You need to verify if this additional segment adversely affects the policy execution in target Cloud Foundry system.

## Migrating Route Service Binding

If you've used the [Managing Cloud Foundry Microservices through API Management \[page 147\]](#) to manage your Cloud Foundry applications, you can now migrate the existing route service binding, from the API Management instance on Neo to the new API Management instance on Cloud Foundry.

### Prerequisites

- A route service binding exists between your application on Cloud Foundry and the API Management service instance in the Neo environment.
- You have enabled API Management on your Cloud Foundry sub account
- You have the space developer role assigned to you.

Depending upon the location of your application, and your API Management service instance, the steps to migrate the route service binding vary.

### Cloud Foundry Application and API Management capability on the same subaccount

If your cloud foundry application and the API Management capability are on the same sub account, then use the following steps to migrate the route service binding:

1. Create an API Management, API portal service instance using the service plan, apim-as-route-service. For more information, see [Creating an API Management, API portal Service Instance \[page 148\]](#)
2. Unbind your application from the API Management service instance on Neo. For more information, see
3. Bind your application to the API Management service instance on Cloud Foundry. For more information, see [Binding a Cloud Foundry Application to an API Management, API portal Service Instance \[page 149\]](#)

### Cloud Foundry Application and API Management capability on different sub accounts

If your Cloud Foundry application and the API Management capability are on different sub accounts, then use the following steps to migrate the route service binding:

1. Create a User Provided Service in the sub account where your Cloud Foundry application is present, using the proxy URL from the sub account in which your API Management instance is present. In order to create this User Provided Service, open the command prompt and use the following command

#### Sample Code

```
cf create-user-provided-service apim-route-service -r https://
apiproxy.url.from.source.
OK
```

For more information, see [User Provided Service](#)

2. Unbind your application from the API Management service instance on Neo. For more information, see
3. Bind the User Provided Service created in the first step to the Cloud Foundry Application. For this binding, use the following command:

#### Sample Code

```
cf bind-route-service cfapps.eu10.hana.ondemand.com --hostname <your-app-host> apim-route-service
```

## Validate Your Target API Management System

Validate that all your API Management artifacts have been cloned to the target system and that all your artifacts and route bindings are in working condition.

## Switch Over from Source to Target System

You can choose to switch over completely from your source to target system after you've successfully cloned all the entities, performed the post-cloning tasks, and validated that your target system is working correctly.

This section explains the various scenarios for a switch-over:

### Switching Over Runtime Proxy URLs

Scenario	Actions Required for Switchover	
If you want to retain the same proxy URL as that of your source system	If the proxy URL of your source system is on a domain managed by SAP	There's no option to retain the old proxy URL.  You must adopt the new proxy URL that is generated for your target system.
	If the proxy URL of your source system is on a custom domain	<ol style="list-style-type: none"> <li>1. Update the virtual host of the target system to that of the source system. See <a href="#">Configuring Additional Virtual Host in Cloud Foundry Environment [page 124]</a>.</li> <li>2. Perform a DNS change from the old cluster to a new cluster.</li> </ol>
If you have multiple virtual hosts configured on your source system subscription, and want to retain those on your target system		<ol style="list-style-type: none"> <li>1. Create multiple virtual hosts on your target system. See <a href="#">Configuring Additional Virtual Host in Cloud Foundry Environment [page 124]</a>.</li> <li>2. Bind each API proxy to the desired virtual host on your target system.</li> </ol>

## Switching Over Design time URLs of API portal and Developer portals

- Domains managed by SAP can't be switched over.
- To switch over a custom domain, create an incident on the component OPU-API-OD-OPS through the [SAP Support Portal](#).

### Applicable Only During Starter Plan Migration

During the switchover, the Tenant Cloning Tool has a wait time of five minutes for the design-time to connect to the runtime on the target API portal. You can enable this feature by setting the "targetDestinationRefreshOnSwitchOver" parameter to true.

Once this parameter is set to true, the wait time is prompted on the console. Use this time to create a test API proxy on the target API portal and try to deploy the same on the virtual host, which is cloned from the source API portal. Once done, key in [Yes](#) on the Tenant Cloning Tool console.

#### Note

When you try to deploy the test proxy on the target, you might encounter an API proxy deployment error because at this point, the connection between the design time and the runtime is still being refreshed. We recommend that you keep trying until the proxy gets deployed successfully. Without a successful deployment, do not proceed with the next steps.

#### Caution

Do not attempt the Stage Switchover of the Tenant Cloning Tool unless it is a "Runtime Reuse Design time Only Migration" scenario.

## 1.15.1.4 Recommendations

This topic lists the recommendations that you must consider for migration.

- After cloning the API Management content from the source to the target system for the first time, you must maintain both the systems, until you switch over from the source system to your target system completely.
- It is recommended that you always add or modify your entities in the source system, and clone it to the target system by rerunning the tool as and when required, instead of adding them directly to the target system.

## 1.15.1.5 Security Features of the Tenant Cloning Tool

The security features of the Tenant Cloning Tool is described in this section.

### Auditing and Logging

- The Tenant Cloning tool calls the APIs provided by and API business hub enterprise(developer portal). Hence, there are no security-related events available in the tool.
- All application logs generated from the cloning tool are stored in "APIM-Tenant-Cloning-Tool.log", an autogenerated log file.

### Data Protection and Privacy

- The tool doesn't persist any data on its own, nor is there a persistence layer.
- The tool logs the cloning status in the log file and in the output excel file named "apim-tct-output.xlsx".
  - The log and output excel file contain e-mail IDs of application developers, needed for troubleshooting and migration reporting, which are being cloned from source to target system.
  - The tool doesn't store any personal data (except e-mail IDs of application developers) in the log and output excel file.
  - We recommend storing the log file and output excel file securely, if further processing is needed; else these files must be deleted.
- The tool doesn't read any sensitive personal data.
- The tool doesn't change any personal data.

### Identity and Access Management

- No specific identity and access management configuration is needed to run the tool.
- Application developer's details are copied from source to target system as is. If some of the developer information isn't valid in the IDP configured in the target tenant, it must be corrected.

### Network and Communication Security

The tool uses standard HTTPS communication to make API calls, as provided by the and API business hub enterprise(developer portal).

## 1.15.1.6 Migrating API Management Subscription Created Using the Starter Plan Service Instance

You can choose to migrate the design-time components that you have in the Neo environment, which was previously set up using Starter Plan instance, to the Cloud Foundry environment, keeping the runtime components as is.

### Context

You can also enable the new API Management design time subscription on the same Cloud Foundry subaccount, where you have created the starter plan service instance.

#### Note

You must subscribe to the API portal and the Developer Portal in the same Cloud Foundry subaccount where the starter plan instance is created.

Tenant type (for example, production and test) of the newly onboarded API Management on the Cloud Foundry environment must be same as that of the source API Management on the Neo environment.

#### Caution

The migration of the Starter Plan Service Instance might involve downtime of the API runtime calls.

### Procedure

1. Raise a ticket through the [SAP Support Portal](#). For more information, see [Product Support](#).

Use the following component for your incident:

Component Name	Component Description
OPU-API-OD-OPS	SAP API Management Operations - On Demand

When submitting the incident, include the following information:

- Incident title: Starter Plan Migration
- Description: State that you want to migrate API Management subscription created using the Starter Plan.
- Provide the Neo account details where API Management is enabled.
- Provide the Cloud Foundry account details where starter plan service instance is created.

#### Note

Once you receive a confirmation from SAP on the ticket, you can resume the migration process from step 2.

2. Prepare the target system by enabling the API Management subscription on the Cloud Foundry subaccount where your starter plan instance was created.

To complete the checks, before you start migrating your API Management artifacts nondisruptively from your source system to a target system, see [Prerequisites \[page 782\]](#).

3. Run the Tenant Cloning Tool in the DEFAULT stage. See [Clone API Management Content \[page 783\]](#) for more information.

For the list of cloned and uncloned entities, see [Cloned and Uncloned Entities \[page 802\]](#). For understanding the behavior of the Tenant Cloning Tool with respect to cloning some of the entities from your source system, see [Tenant Cloning Tool Behavior \[page 804\]](#).

#### Note

Since this is starter plan migration scenario, only the API portal artifacts at this stage get cloned.

4. After completing the cloning process, you must perform some actions, checks, and validations. For the task details, see [Post Cloning Tasks \[page 805\]](#).

For recommendations for migration, refer the following topic: [Recommendations \[page 811\]](#)

To know more about the security features of the tenant cloning tool, see [Security Features of the Tenant Cloning Tool \[page 812\]](#).

5. Run the Tenant Cloning Tool in the SWITCHOVER stage. For more information, see [Clone API Management Content \[page 783\]](#).

#### Note

If applicable, API business hub enterprise (developer portal) entities are cloned in this step.

6. After the SWITCHOVER, if you have any API Provider of the type onpremise, provide the Basic Auth password in the target system. For more information, see "API Provider Credentials" under [User Actions in Post Cloning Tasks \[page 805\]](#).

#### Note

If you skip this step, the test connection on the particular on-prem provider will fail. Also, the discovery of the on-prem providers will fail. Therefore, please ensure that you complete this step before proceeding.

7. Inform SAP that migration is complete by updating the same ticket.

#### Note

If you encounter any issues during the migration process, you can report and track the updates in the same ticket. Therefore, we recommend that you keep the ticket open until you reach step 6.

## Results

Migration of API Management subscription created using the Starter Plan service instance is complete. There can be downtime for certain API proxies (having policies that are specific to Neo/ Cloud Foundry environment) created out of on-premise providers.

## 1.15.1.7 Migrating API Management Subscription Created Using the Starter Plan Service Instance to Different Subaccounts

Migrate the design-time components from the Neo environment, which was previously set up using Starter Plan instance, to the Cloud Foundry environment, keeping the runtime components as is.

### Context

With the Integration Suite premium edition license available in a different subaccount, you can migrate the API Management design time subscription to this subaccount as well.

#### Note

Make a note of the following:

- Analytics data can't be retained, as Advanced Analytics gets newly configured in the different subaccount. However, if you come across any analytics data from the previous subaccount, you must ignore the data and consider the analytics data after the migration task is completed.
- Subscribe to the API portal and the API business hub enterprise in the other Cloud Foundry subaccount. This is the subaccount with the Integration Suite premium edition license.
- Tenant type (for example, production and test) of the newly onboarded API Management on the Cloud Foundry environment must be same as that of the source API Management on the Neo environment.
- Ensure that both the source and the target subaccounts are in the same data center for migrating the API Management subscription to a different subaccount.
- To migrate to a different subaccount, you must provide the input `cfSubaccountTenantID` in the `apim-tct-input.json` file. For more information, see [Clone API Management Content \[page 783\]](#).

#### Caution

The migration of the Starter Plan Service Instance may involve downtime of the API runtime calls.

### Procedure

1. Raise a ticket through the [SAP Support Portal](#). For more information, see [Product Support](#).

Use the following component for your incident:

Component Name	Component Description
OPU-API-OD-OPS	SAP API Management Operations - On Demand

When submitting the incident, include the following information:

- Incident title: Starter Plan Migration to Different Subaccounts

- Description: State that you want to migrate API Management subscription created using the Starter Plan to different subaccounts.
- Provide the Neo account details where API Management is enabled.
- Provide the Cloud Foundry account details where starter plan service instance is created.
- Provide the details of the target Integration Suite subscription account for migrating the starter plan subscription to a different subaccount. Since you already have the Integration Suite premium license available, you can create the API Management subscription before creating the ticket.

#### Note

Once you receive a confirmation from SAP on the ticket, you can resume the migration process from step 2.

2. To avoid any disruption, complete the checks before you start migrating your API Management artifacts from your source system to your target system. For details, see the [Prerequisites \[page 782\]](#) section.
3. Run the Tenant Cloning Tool in the DEFAULT stage. See [Clone API Management Content \[page 783\]](#) for more information.

For the list of cloned and uncloned entities, see [Cloned and Uncloned Entities \[page 802\]](#). For understanding the behavior of the Tenant Cloning Tool with respect to cloning some of the entities from your source system, see [Tenant Cloning Tool Behavior \[page 804\]](#).

#### Note

Since this is starter plan migration scenario, only the API portal artifacts at this stage get cloned.

4. After completing the cloning process, you must perform some actions, checks, and validations. For the task details, see [Post Cloning Tasks \[page 805\]](#).

For recommendations for migration, refer the following topic: [Recommendations \[page 811\]](#)

To know more about the security features of the tenant cloning tool, see [Security Features of the Tenant Cloning Tool \[page 812\]](#).

5. Run the Tenant Cloning Tool in the SWITCHOVER stage. For more information, see [Clone API Management Content \[page 783\]](#).

#### Note

If applicable, API business hub enterprise entities are cloned in this step.

6. After the SWITCHOVER, if you have any API Provider of the type onpremise, provide the Basic Auth password in the target system. For more information, see "API Provider Credentials" under [User Actions in Post Cloning Tasks \[page 805\]](#).

#### Note

If you skip this step, the test connection on the particular on-prem provider will fail. Also, the discovery of the on-prem providers will fail. Therefore, please ensure that you complete this step before proceeding.

7. Inform SAP that migration is complete by updating the same ticket.



### Note

If you encounter any issues during the migration process, you can report and track the updates in the same ticket. Therefore, we recommend that you keep the ticket open until you reach step 6.

## Results

Migration of API Management subscription (created using the Starter Plan service instance) to a different subaccount is complete. There can be downtime for certain API proxies (having policies that are specific to Neo/ Cloud Foundry environment) created out of on-premise providers.

## 1.15.2 Migration of API Management Content between Cloud Foundry Environments

You have the option to migrate your API Management content from one Cloud Foundry environment to another. This migration is possible between tenants within the same data center or between tenants located in different data centers.

### Note

Runtime re-use of the tenant is not yet supported.

At the end of the Cloud Foundry to Cloud Foundry migration, your content from source Cloud Foundry is cloned to the target Cloud Foundry.

Migration Assistant for asset migration includes the tools and utilities that enable migration of design time assets non-disruptively from the Cloud Foundry to the Cloud Foundry environment.

Your source system is the system that has your API Management content in the Cloud Foundry environment.

Your target system is the system that has your API Management content within the Cloud Foundry environment. Here Cloud Foundry environment can be your native standalone API Management subscription or API Management capability within Integration Suite.

Possible Migration Paths

Source Subscription	Target Subscription	Allowed
Standalone	Standalone	Yes
Standalone	Integration Suite	Yes
Integration Suite	Standalone	No
Integration Suite	Integration Suite	Yes

The target system may or may not retain the application key/secret based on the target system runtime cluster. This can be pre-checked with OPS via ticket during initial steps.

If the target can't retain the application key/secret then the applications will be created with a new application key/secret in the target system. You must guide your end users to adopt to the change in application key/secret.

Source and target subscriptions must be in same environment. Both the source and the target subscription must be in production environment or both must be in non-production environment. Cross environment migration is not supported.

You can clone your API Management content non-disruptively from the source to the target system only after completing the steps in the prerequisites section. Post cloning, you must complete some user actions and validate your target system.

### Note

The developer portal is renamed to API business hub enterprise in Cloud Foundry environment. In this document API business hub enterprise is referred to as developer portal even in Cloud Foundry environment.

The steps assisting the migration of your API Management from your source system to a target system are:

1. Raise a ticket through the [SAP Support Portal](#). For more information, see [Product Support](#). Use the following component for your incident:

Component Name	Component Description
OPU-API-OD-OPS	SAP API Management Operations - On Demand

When submitting the incident, include the following information:

- Incident title: API Management Migration from one Cloud Foundry to another Cloud Foundry environment
- Description: State that you want to migrate API Management subscription from one Cloud Foundry to another Cloud Foundry environment .
- Provide the Cloud Foundry account details where API Management is enabled.
- Provide the Cloud Foundry account details where you want to move the data .

### Note

Once you receive a confirmation from SAP on the ticket, you can resume the migration process from step 2.

2. Complete all the steps in the **Prerequisite** section.
3. Clone the API Management content using the Tenant Cloning tool.
4. Complete the post cloning tasks.

## 1.15.2.1 Prerequisites for the Source and the Target System

Checks to be completed before you start migrating your API Management content nondisruptively from your source system to a target system.

Both the source and the target system are the system that has your API Management content on the hyperscalers-managed infrastructure within the Cloud Foundry environment.

## Prerequisites for the source system

- You must have a valid API Management system ( and API business hub enterprise) running in the Cloud Foundry environment.
- The source system must support OAuth client credentials for Cloud Foundry. You need the auth token url and key secret to access the and API business hub enterprise. For more information, refer [Accessing API Management APIs Programmatically](#)[Accessing API business hub enterprise APIs Programmatically](#).
- Make a note of the API portal and API business hub enterprise URLs of the source system and keep handy.
- Ensure that API access is enabled for the and API business hub enterprise systems for the following roles:
  - APIPortal.Administrator
  - AuthGroup.API.Admin

The client id, client secret are used while filling in the details of the `apim-tct-input.json` file before running the Tenant Cloning Tool. See [Clone API Management Content \[page 783\]](#).

### Note

During Cloud Foundry to Cloud Foundry migration, the default `input.json` shipped with Tenant Cloning Tool maven zip bundle has username and password as the source field. Please ensure that you change this to token URL, clientId, and client secret as mentioned in the sample configuration in [Clone API Management Content for Cloud Foundry to Cloud Foundry Migration \[page 820\]](#).

## Prerequisites for the target system

- If API Management is not already enabled on your target system, complete the set-up. For more information, see [Initial Setup](#) and [Enable API Management Capability](#) . Check whether the API Management service broker service instance is created with the Starter Plan in the same subaccount.

Service Instance for Starter Plan in the Subaccount	Instruction
Already present	<p>You can't use this subaccount for migration. Create a new subaccount in the hyperscalers-managed infrastructure within the cloud foundry environment and enable API Management on that subaccount for it to act as your target system.</p> <p>Additionally, if you want to reuse the existing runtime then follow the steps mentioned in the <a href="#">Migrating API Management Subscription Created Using the Starter Plan Service Instance [page 813]</a>.</p>
Not present	<p>You can choose to reuse this account as your target system for migration, or create a new subaccount.</p>

You can also consider the "Possible Migration Paths" table in [Migration of API Management Content between Cloud Foundry Environments \[page 817\]](#) to choose the target subscription type.

If you have already enabled API Management on your target system, and want to reuse the same for migration, you can refer the following recommendations:

- It's recommended that you don't have any pre-existing entities such as API proxies or products on this system.

### Note

Any entity, if pre-existing in your target API Management capability, can be over-written during the cloning process.

- If your target system is connected to a custom IDP, ensure that your IDP is configured correctly, and mapping for the details like your first name, last name, email ID, and user ID is done.

### Note

Please ensure that the application developer's attributes, like first name, last name, email ID, and user ID, are identical in both source and target identity providers. In API Management, the application developer's attributes are case-sensitive.

Consider the following example: During cloning, the email address `john.smith@abc.com` in the source becomes `John.Smith@abc.com` in target due to the change in configurations in Custom IDP. This mismatch might lead to data discrepancy during application creation and metering in the target after cloning.

- Ensure that API access is enabled for the and the API business hub enterprise for the following roles:
  - APIPortal.Administrator
  - AuthGroup.API.Admin

Make a note of the service keys (`url`, `tokenurl`, `clientId`, and `clientSecret`) for the given roles, and keep handy. To know more about API access plans for API portal, see [Accessing API Management APIs Programmatically](#). To know more about API access plan for API business hub enterprise, see [Accessing API business hub enterprise APIs Programmatically](#), without which the cloning of the API business hub enterprise entities might fail.

- When you have API products protected by the custom roles permission in the source Cloud Foundry system, ensure that custom roles creation and assignments are done in the target Cloud Foundry environment before starting the migration.

Once you complete these checks, you can start cloning the API Management content from the source to the target system. See [Clone API Management Content for Cloud Foundry to Cloud Foundry Migration \[page 820\]](#).

## 1.15.2.2 Clone API Management Content for Cloud Foundry to Cloud Foundry Migration

Clone the API Management content using the Tenant Cloning Tool.

Once you have your source and target system ready, you can clone your API Management content to the target system by running the Tenant Cloning Tool that you downloaded from [here](#).

### Prerequisites

- You must have downloaded the Tenant Cloning Tool ( ) from the link provided above. APIM-TCT-  
<version>.zip

- APIM-TCT-You must have extracted the contents of the `APIM-TCT-<version>.zip` file into a folder (example name `apim-tct`).

This extracted folder must contain:

- a `java apim-tct-client-<version>.jar` file
- a sample `apim-tct-input.json` file
- a `lib` folder (this folder and its contents must not be modified)
- a `README.md` file
- Script files to download open-source libraries that are required to run the `apim-tct-client-<version>.jar` file:
  - `download_dependencies.ps1` for Windows systems
  - `download_dependencies.sh` for Mac and Linux systems

#### Note

Download the dependencies as described in the **Downloading the Dependencies** section.

#### Note

If you are using the version of the Tenant Cloning Tool prior to 1.5.2, make sure that you update to the latest version 1.5.2 or above. This is done to handle the critical vulnerability CVE-2021-44228 and CVE-2021-45046, which was detected in the open-source library log4j2.

- The system running the API Management Tenant Cloning Tool must have Java Runtime Environment 8 or above supported.
- Microsoft Excel File Reader

## Downloading the Dependencies

### For Windows Systems:

- Open the PowerShell terminal.
- Go to the `apim-tct` folder in the terminal.
- Run the `.\download_dependencies.ps1` command.  
The required libraries are downloaded to the `lib` folder.

### For Mac and Linux Systems:

- Open the default terminal from your system.
- Go to the `apim-tct` folder in the terminal.
- Run the `chmod +x download_dependencies.sh` command to make the file executable.
- Run the `.\download_dependencies.sh` command.  
The required libraries are downloaded to the `lib` folder.

#### Note

If you encounter an error while running these commands, then you can download the dependencies manually from the link provided in the script file and place them into the `lib` folder.

## Procedure

1. Fill in the `apim-tct-input.json` file by providing details such as the URLs of your source and target systems, access token URLs, client id, and client secret to your source and target systems. Ensure that you don't modify the name of the `apim-tct-input.json` file. For more information on how to create the service key, refer the [Accessing API Management APIs Programmatically \[page 127\]](#) and [Accessing API business hub enterprise APIs Programmatically \[page 134\]](#).

Structure of the `apim-tct-input.json` file:

Input Field	Type of Credentials	Data Type	Required/Optional	Description	
source	apiportal	url	String	Required	URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role
<div style="border-left: 2px solid #0070C0; padding-left: 10px; margin-bottom: 10px;"> <p><b>Note</b></p> <p>Choose the relevant fields based on the credential type you've configured for the API access plan. For example, if you've used Client Secret as the credential type, do not select the fields from X509 mTLS.</p> </div>					
	tokenUrl	Client Secret	String	Optional	Token URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role

Input Field	Type of Credentials	Data Type	Required/Optional	Description
<code>clientId</code>		String	Optional	<p>The client ID received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p> <p>To know more about creating the service key, see <a href="#">Accessing API Management APIs Programmatically [page 127]</a>.</p>

Input Field	Type of Credentials	Data Type	Required/Optional	Description
<code>clientSecret</code>		String	Optional	<p>The client secret received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>
<code>certurl</code>	X509 mTLS	String	Required	Cert URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role.
<code>certificate</code>		String	Required	The content of the certificate received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role.



Input Field	Type of Credentials	Data Type	Required/Optional	Description	
	<b>clientid</b>	String	Required	Client ID received during creation of the service key for API portal API access for the APIPortal.Administrator role.	
	<b>privatekey</b>	String	Required	Private Key received during creation of the service key for API portal API access for the APIPortal.Administrator role.	
devportal	<b>url</b>	String	Required	URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role	
	<b>tokenUrl</b>	Client Secret	String	Optional	Token URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role

**Note**

Choose the relevant fields based on the credential type you've configured for the API access plan. For example, if you've used Client Secret as the credential type, do not select the fields from X509 mTLS.

Input Field	Type of Credentials	Data Type	Required/Optional	Description
<code>clientId</code>		String	Optional	<p>The client ID received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>
<code>clientSecret</code>		String	Optional	<p>The client secret received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>

Input Field	Type of Credentials	Data Type	Required/Optional	Description
<code>certurl</code>	X509 mTLS	String	Optional	Certificate URL received during creation of the service key for API portal API access for the APIPortal.Administrator role.
<code>certificate</code>		String	Optional	The contents of the certificate received during creation of the service key for API portal API access for the APIPortal.Administrator role.
<code>clientid</code>		String	Optional	Client ID received during creation of the service key for API portal API access for the APIPortal.Administrator role.
<code>privatekey</code>		String	Optional	Private Key received during creation of the service key for API portal API access for the APIPortal.Administrator role.
cfSubaccount-TenantID	Supported values: "guid"	String	Not Applicable	

Input Field		Type of Credentials	Data Type	Required/Optional	Description	
target	apiportal	<b>Url</b>	String	Required	URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role	
<p><b>Note</b></p> <p>Choose the relevant fields based on the credential type you've configured for the API access plan. For example, if you've used Client Secret as the credential type, do not select the fields from X509 mTLS.</p>		<b>tokenUrl</b>	Client Secret	String	Required	Token URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role
		<b>clientId</b>		String	Optional	The client ID received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role
						You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.

Input Field	Type of Credentials	Data Type	Required/Optional	Description
<code>clientSecret</code>		String	Optional	<p>The client secret received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>
<code>certurl</code>	X509 mTLS	String	Optional	Certificate URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role.
<code>certificate</code>		String	Optional	The contents of the certificate received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role.

Input Field	Type of Credentials	Data Type	Required/Optional	Description
	<code>clientid</code>	String	Optional	Client ID received during creation of the service key for API portal API access for the APIPortal.Administrator role.
	<code>privatekey</code>	String	Optional	Private Key received during creation of the service key for API portal API access for the APIPortal.Administrator role.
devportal	<code>url</code>	String	Required	URL received during creation of the service key for API business hub enterprise API access for the <b>AuthGroup.API.Admin</b> role.
	<code>tokenUrl</code>	String	Required	Token url received during creation of the service key for API business hub enterprise API access for the <b>AuthGroup.API.Admin</b> role.

**Note**

Choose the relevant fields based on the credential type you've configured for the API access plan. For example, if you've used Client Secret as the credential type, do not select the fields from X509 mTLS.

Input Field	Type of Credentials	Data Type	Required/Optional	Description
	<code>clientId</code>	String	Optional	<p>The client ID received during creation of the service key for API business hub enterprise API access for the <b>AuthGroup.API.Admin</b> role.</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>

Input Field	Type of Credentials	Data Type	Required/Optional	Description
<code>clientSecret</code>		String	Optional	<p>The client secret received during creation of the service key for API business hub enterprise API access for the <b>AuthGroup.API.Admin</b> role.</p> <p>You're prompted to enter these values while running the command in Step 3 if you haven't already provided these details in the <code>apim-tct-input.json</code> file.</p>
<code>certurl</code>	X509 mTLS	String	Optional	Certificate URL received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role.
<code>certificate</code>		String	Optional	The contents of the certificate received during creation of the service key for API portal API access for the <b>APIPortal.Administrator</b> role.



Input Field	Type of Credentials	Data Type	Required/Optional	Description
<code>clientid</code>		String	Optional	Client ID received during creation of the service key for API portal API access for the APIPortal.Administrator role.
<code>privatekey</code>		String	Optional	Private Key received during creation of the service key for API portal API access for the APIPortal.Administrator role.

Input Field	Type of Credentials	Data Type	Required/Optional	Description
skipApplicationKeySecretCloning	Supported values: <b>true</b> / <b>false</b>	Boolean	Optional	<ul style="list-style-type: none"> <li>The default value for skipApplicationKeySecretCloning is false.</li> </ul>

**Note**

If you want to skip the cloning of Application Key and Secret in side by side migration, then set the `"skipApplicationKeySecretCloning"` flag to true.

Input Field	Type of Credentials	Data Type	Required/Optional	Description
				<p>This value is determined as per the inputs from the Operations team on the ticket raised . If you don't follow the recommendations from the Operations team the cloning of the application on the target might fail.</p>

Input Field		Type of Credentials	Data Type	Required/Optional	Description
clone	skip-apiportal	Supported values: <b>true/false</b>	Boolean	Optional	<ul style="list-style-type: none"> <li>The default value for skip-apiportal is false, and API portal entities are cloned</li> <li>If you set the value for skip-apiportal to true, no cloning of the API portal entities takes place.</li> </ul>
	skip-devportal	Supported values: <b>true/false</b>	Boolean	Optional	<ul style="list-style-type: none"> <li>The default value for skip-devportal is false, and API business hub enterprise entities are cloned.</li> <li>If you set the value for skip-devportal to true, no cloning of the API business hub enterprise entities takes place.</li> </ul>

Input Field	Type of Credentials	Data Type	Required/Optional	Description
stage	Supported values: <b>"DEFAULT"</b>   <b>"SWITCHOVER"</b> <b>R</b>	string	Optional	The default values for this parameter is supported. Switch-over is not applicable for this scenario.

\*\*\* API portal credentials for source and target for all scenarios are mandatory.

### → Remember

For the clone input attribute:

- Both skip-apiportal and skip-devportal are set to false by default, so, API portal entities are cloned first, followed by API business hub enterprise entities.
- If both skip-apiportal and skip-devportal are set to true, no cloning takes place.
- If skip-apiportal is set to false, but skip-devportal is set to true, then only the API portal entities are cloned.
- If skip-apiportal is set to true, but skip-devportal to false, then only API business hub enterprise entities are cloned and cloning for entities (like applications) may fail, pertaining to nonavailability of dependent entity (like API Product) in API business hub enterprise.

Sample configuration:

```
{
 "source": {
 "apiportal": {
 "url": "<URL of Source (Cloud Foundry based) API Portal>",
 "tokenUrl": "<token url received during service key creation for API Portal's API Access for APIPortal.Administrator role. For example, https://<Space name>.authentication.sap.hana.ondemand.com/oauth/token>",
 "clientId": "<clientId received during service key creation for API Portal's API Access for APIPortal.Administrator role. For example, sb-apiaccesssxxxxxxxx!xxxx|api-portal-xsuaa!bxxxx>",
 "clientSecret": "<clientSecret received during service key creation for API Portal's API Access for APIPortal.Administrator role>"
 },
 "devportal": {
 "url": "<URL of Source (Cloud Foundry based) API business hub enterprise>",
 "tokenUrl": "<token url received during service key creation for API business hub enterprise's API Access for AuthGroup.API.Admin role. For example, https://<Space name>.authentication.sap.hana.ondemand.com/oauth/token>",
 "clientId": "<clientId received during service key creation for API business hub enterprise's API Access for AuthGroup.API.Admin role. For example, sb-apiaccesssxxxxxxxx!xxxx|api-portal-xsuaa!bxxxx>",
 "clientSecret": "<clientSecret received during service key creation for API business hub enterprise's API Access for AuthGroup.API.Admin role>"
 }
 },
 "target": {
 "apiportal": {
 "url": "<URL of Source (Cloud Foundry based) API Portal>",
```

```

 "tokenUrl": "<token url received during service key creation for
API Portal's API Access for APIPortal.Administrator role>",
 "clientId": "<clientId received during service key creation for
API Portal's API Access for APIPortal.Administrator role>",
 "clientSecret": "<clientSecret received during service key
creation for API Portal's API Access for APIPortal.Administrator role>"
 },
 "devportal": {
 "url": "<URL of Source (Cloud Foundry based) API business hub
enterprise>",
 "tokenUrl": "<token url received during service key creation for
API business hub enterprise's API Access for AuthGroup.API.Admin role>",
 "clientId": "<clientId received during service key creation for
API business hub enterprise's API Access for AuthGroup.API.Admin role>",
 "clientSecret": "<clientSecret received during service key
creation for API business hub enterprise's API Access for AuthGroup.API.Admin
role>"
 }
},
"skipApplicationKeySecretCloning" : <false|true>,

"clone": {
 "skip-apiportal": <false|true> ,
 "skip-devportal": <false|true>
},
"stage": <"DEFAULT">
}

```

2. Run the following commands from your Java command-line interface to verify the setup and check the version of the tool. This is an optional step.
  - To verify the setup:

```
java -jar apim-tct-client-<version>.jar verify
```
  - To check the version of the tenant cloning tool you're using:

```
java -jar apim-tct-client-<version>.jar version
```
3. To begin the cloning process, run the following command from your Java command-line interface:

```
java -jar apim-tct-client-<version>.jar
```

#### Result

Your API Management entities are now cloned to your target system.

An excel file named `apimtct-output.xlsx` and a log file named `apimtct-logs.log` are generated in the same folder where the `.jar` file is present.

The status of each cloned entity is stored in a separate worksheet within the output excel file.

Structure of a Worksheet Within `apimtct-output.xlsx` File

Column	Description
ID	Entity ID
Name	Entity name
Type	Entity type
Script Execution Timestamp (UTC)	Script execution time in UTC
Artifact's Last Modified Timestamp (UTC)	Last modified time of the entity in the source API Management system (UTC)

Column	Description
STATUS	Migration Status: <ul style="list-style-type: none"> <li>• SUCCESS (Entity successfully cloned)</li> <li>• FAILURE (Entity failed to clone)</li> <li>• SKIPPED (Cloning of Entity skipped)</li> </ul>

You can view the status of the cloned content in the `apimct-output.xlsx` file or in the `apimct-logs.log` file.

#### Note

- Ensure that the `apimct-output.xlsx` file isn't open while you run the script.
- It's recommended that you don't modify the `apimct-output.xlsx` file.

#### Troubleshooting During Cloning:

- If the Tenant Cloning Tool shuts down unexpectedly, restart and try again. If the tool throws an error repeatedly while running, you can report the incident or error on the component OPU-API-OD-DT through the [SAP Support Portal](#).

## Next Steps

After the cloning process completes, you must perform the tasks mentioned in the **User Actions** worksheet within the output excel file `apimct-output.xlsx`.

To know more about what actions you must take, see the **User Actions** section in [Post Cloning Tasks \[page 843\]](#).

To know more about the entities that are cloned and the entities that aren't cloned, see [Cloned and Uncloned Entities \[page 839\]](#).

### 1.15.2.2.1 Cloned and Uncloned Entities

Refer this section for the entities that are cloned and entities that aren't cloned during the migration process.

#### Entities That Are Cloned

##### Note

Currently, when a custom role is assigned to a product, the application creation using the tenant cloning tool is not supported.

As a work-around, before initiating the cloning process, remove the custom role assigned to the product in the source system and proceed with the cloning process.

After the cloning process is completed, reassign the custom roles to the product in the source system. Also, ensure that the custom roles are assigned to the product in the target system.

In case the custom roles aren't appearing in the *Permission* tab, as mentioned in the prerequisite section, ensure that the custom roles are created and assigned to the developers in the target Cloud Foundry environment.

## Note

If you have made any customizations to the `HelloWorld` sample proxy, and you want to migrate this proxy to the target, while cloning you might get the following error: "Unable to import API Proxy from zip file; xml content invalid". To address this, execute the following steps:

1. Export the `HelloWorld` API.
2. Open the zip file and edit the `metadata.xml` file to add the `created_by` field as shown below:

### Sample Code

```
<life_cycle>
 <changed_by>yourUserId</changed_by>
 <created_by>yourUserID</created_by>
</life_cycle>
```

Please note that your `userId` is as per your Identity Service configuration. You can find your `userId` when you open any proxies in the API portal.

3. Save the zip file.
4. Delete the existing `HelloWorld` proxy from .
5. Import this edited zip file.

With this the `created_by` will reflect in the API proxy.

The following list displays the API Management entities that are cloned:

- Certificates and Certificate Store
- Rate Plans
- Key Value Maps
- API Providers
- Policy Templates
- API Proxies
- API Products
- Measure Codes for Custom Measures
- Dimension Codes for Custom Dimensions
- Application
- Application Developer
- Access Control Permissions for API Product
- Custom Metrics and Charts



## Content That Are Not Cloned

The following list displays the API Management content that aren't cloned, including sensitive data like your certificates and credentials.

- **Sensitive Data**

- Certificates
- Encrypted Key Value Maps
- API Provider Passwords
- Monetization Bills

To know about the actions that you must perform for the uncloned certificates, encrypted key value maps, and API provider passwords, see the **User Actions** section in [Post Cloning Tasks \[page 843\]](#).

- **Runtime data**

- Quota Counters
- OAuth Tokens for API Proxy runtime calls
- Runtime states of any API Management entity

- **Configurations**

- Cloud Connector Setup
- Custom Role creation and its assignments
- Default role assignment to users
- Principal Propagation setup for OpProxy
- Any configurations created at the subaccount level
- Any integrations with other systems (like SAP Web IDE)
- Custom IDP Setup (if any)
- Existing Route Bindings (if any)

To know about the actions that you must perform for these uncloned content, see the **Actions required on Configurations** section in [Post Cloning Tasks \[page 805\]](#).

### 1.15.2.2.2 Tenant Cloning Tool Behavior

This topic describes the behavior of the Tenant Cloning Tool with respect to cloning some of the entities from your source system.

- If you add or modify an entity in your source system, it is always cloned to the target system in your subsequent run of the Tenant Cloning Tool.
- If you add a new entity to your target system at any point, it is retained in the target system after the subsequent run of the Tenant Cloning Tool, irrespective of whether the entity is present in your source system or not.
- Newer state of an existing entity present in your source system is always migrated to the target system after the subsequent run of the Tenant Cloning Tool, and overwrites any older state of the entity in the target.
- During the cloning of the Developer Portal entity **Application Developer**, the app developer receives email notifications while being onboarded to the target Developer Portal.  
We recommend that you inform your developers about the impending migration and email notifications that they might receive during the process.

- Custom Charts are cloned to the target as many times as you run the Tenant Cloning Tool.
- All the API proxies are cloned onto the default virtual host.
- Post cloning, the API proxies on the target system are in active and deployed state. You must reapply the desired states to the proxies.

To know more about API proxy states, see [API Proxy States \[page 580\]](#).

#### Note

If the Tenant Cloning Tool is used to clone an API proxy or a product with more than 100 resources attached to it, you might notice data inconsistency in the target system (API business hub enterprise or ). It is recommended that you do not add more than 100 resources per proxy or product. For more information, see [Limits in API Management \[page 776\]](#).

- Cloning of custom chart is now supported for migrating API Management content created using the Starter Plan service instance.

#### Note

Known constraints of migrating applications:

##### Use Case 1:

You are migrating application A1, subscribed to a product P1 with no rate plans. You associate product P1 with a rate plan later in the source.

Now if you attempt to migrate application A1 from the source developer portal to the target developer portal, the application created in the target fails with an error.

**Reason for the error:** During migration, the product gets cloned along with the newly added rate plan in the target tenant. However, while creating the application in the target tenant, the system couldn't locate the rate plan ID. The rate plan ID was not present in the payload as it belonged to an older subscription, resulting in application creation failure.

##### Use Case 2:

You already have a rate plan attached to a product P1 while creating application A1. However, you decide to add a revised rate plan to product P1 after application A1 gets created. In such a scenario, application A1 will continue to have a rate plan that was present during the creation of the application. If you migrate application A1 from the source developer portal to the target developer portal, application creation will fail, and the same error as Use Case 1 will get generated.

**Reason for the error:** During migration, the product gets cloned along with the newly revised rate plan in the target tenant. However, while creating the application in the target tenant, the system couldn't locate the revised rate plan ID. The revised rate plan ID was not present in the payload as it belonged to an older subscription, resulting in application creation failure.

#### Note

CacheResources cloning is currently not supported via the Tenant Cloning Tool. You must create it manually on the target using the following service: `<apiportal-host>/apiportal/api/1.0/Management.svc/CacheResources`.

## 1.15.2.3 Post Cloning Tasks

Post the completion of the cloning process, you must perform some actions, checks, and validations.

The following sections explain the tasks that you must perform after the cloning of your API Management artifacts from the Cloud Foundry to the Cloud Foundry environment is complete.

### User Actions

You can view the status of the cloned artifacts in the `apim-tct-output.xlsx` excel file or in the `apim-tct.log` file, generated in the same folder where the `.jar` file is present.

Perform the tasks mentioned in the **User Actions** worksheet within the `apim-tct-output.xlsx` excel file.

The following table describes the actions required for each cloned entity:

Cloned Entity	User Action
Certificates	<p>All the certificates that are cloned to the target system are dummy certificates.</p> <p>Perform the following steps:</p> <ol style="list-style-type: none"><li>1. From your source system, note down the certificate names and the corresponding certificate store name.</li><li>2. From your target system, delete the dummy certificates that were cloned:<ol style="list-style-type: none"><li>1. In your target API Management, API portal, navigate to <b>Configure &gt; Certificates</b>.</li><li>2. Select the cloned dummy certificate that you want to delete.</li><li>3. Click the delete icon under the <i>Actions</i> column.</li></ol></li><li>3. In your target API Management, API portal, upload the relevant certificates, providing the same names and under same certificate store as present in your source system.<ol style="list-style-type: none"><li>1. In your target API Management, API portal, navigate to <b>Configure &gt; Certificates</b>.</li><li>2. Click <i>Create</i>.</li><li>3. In the <i>Create Certificate</i> window, provide the details and upload the certificate.</li></ol></li></ol>
Key Value Maps	<p>Fill in the values for the keys of the encrypted Key Value Maps.</p> <ol style="list-style-type: none"><li>1. In your target API Management, API portal, navigate to <b>Configure &gt; Key Value Maps</b>.</li><li>2. Click on the encrypted key value map.</li><li>3. In the <i>Edit Key Value Map</i> window, provide the details and click <i>Save</i>.</li></ol>

## Cloned Entity

## User Action

API Provider Credentials

Provide the Basic Auth password (if present in your source system) for an API Provider.

1. In your target API Management, API portal, navigate to [Configure > API Providers](#).
2. Click on the desired API provider.
3. In the *View API Provider* window, click [Catalog Service Settings > Edit](#).
4. Provide the basic auth password for the API provider and click *Save*.

Update open connector credentials:

1. In your target API Management, API portal, navigate to [Configure > API Providers](#).
2. Click on the desired API provider.
3. In the *View API Provider* window, click [Catalog Service Settings > Edit](#).
4. Provide the Org ID and User Secret from your corresponding Open Connector subscription.

Proxy Scoped Key Value Map

Provide instance token value in the proxy scoped Key Value Map.

1. In your target API Management, API portal, navigate to [Develop > APIs](#).
2. Click on the desired API proxy.
3. In the *View API* page, scroll down to *Key Value Map Associated* and choose the Key Value Map.
4. On the *Edit Key Value Map* page, update the *Value* with the instance token value for the corresponding open connector instance.
5. Provide the Org ID and User Secret from your corresponding Open Connector subscription.

## Actions Required on Configurations

Depending on the configurations you have on your source system, you must configure the following in your target system:

- Custom IDP Setup (if any)
- Default role assignment to users
- Custom Role creation and its assignments
- Cloud Connector setup

- Principal Propagation setup at the subaccount level
- Changes to Principal Propagation policy for on-premise connectivity
- Migration of route service bindings. For more information, see [Migrating Route Service Binding \[page 809\]](#)
- Any integrations with other systems (like SAP Web IDE)
- Any other configurations that you created for API Management at the subaccount level of your source system

To know more about the entities that are cloned and the entities that aren't cloned, see [Cloned and Uncloned Entities \[page 839\]](#).

## Migrating Route Service Binding

If you've used the [Managing Cloud Foundry Microservices through API Management \[page 147\]](#) to manage your Cloud Foundry applications, you can now migrate the existing route service binding, from the API Management instance on Cloud Foundry to the new API Management instance on Cloud Foundry.

### Prerequisites

- A route service binding exists between your application on Cloud Foundry and the API Management service instance in the Cloud Foundry environment.
- You have enabled API Management on your Cloud Foundry sub account
- You have the space developer role assigned to you.

Depending upon the location of your application, and your API Management service instance, the steps to migrate the route service binding vary.

### Cloud Foundry Application and API Management capability on the same subaccount

If your cloud foundry application and the API Management capability are on the same sub account, then use the following steps to migrate the route service binding:

1. Create an API Management, API portal service instance using the service plan, apim-as-route-service. For more information, see [Creating an API Management, API portal Service Instance \[page 148\]](#)
2. Unbind your application from the API Management service instance on Cloud Foundry. For more information, see
3. Bind your application to the API Management service instance on Cloud Foundry. For more information, see [Binding a Cloud Foundry Application to an API Management, API portal Service Instance \[page 149\]](#)

### Cloud Foundry Application and API Management capability on different sub accounts

If your Cloud Foundry application and the API Management capability are on different sub accounts, then use the following steps to migrate the route service binding:

1. Create a User Provided Service in the sub account where your Cloud Foundry application is present, using the proxy URL from the sub account in which your API Management instance is present. In order to create this User Provided Service, open the command prompt and use the following command

#### Sample Code

```
cf create-user-provided-service apim-route-service -r https://
apiproxy.url.from.source.
```

OK

For more information, see [User Provided Service](#) 

2. Unbind your application from the API Management service instance on Cloud Foundry. For more information, see
3. Bind the User Provided Service created in the first step to the Cloud Foundry Application. For this binding, use the following command:

#### Sample Code

```
cf bind-route-service cfapps.eu10.hana.ondemand.com --hostname <your-app-host> apim-route-service
```

## Validate Your Target API Management System

Validate that all your API Management artifacts have been cloned to the target system and that all your artifacts and route bindings are in working condition.

## Switch Over from Source to Target System

You can choose to switch over completely from your source to target system after you've successfully cloned all the entities, performed the post-cloning tasks, and validated that your target system is working correctly.

This section explains the various scenarios for a switch-over:

### Switching Over Runtime Proxy URLs

Scenario	Actions Required for Switchover	
If you want to retain the same proxy URL as that of your source system	If the proxy URL of your source system is on a domain managed by SAP	There's no option to retain the old proxy URL.  You must adopt the new proxy URL that is generated for your target system.
	If the proxy URL of your source system is on a custom domain	<ol style="list-style-type: none"><li>1. Update the virtual host of the target system to that of the source system. See <a href="#">Configuring Additional Virtual Host in Cloud Foundry Environment [page 124]</a>.</li><li>2. Perform a DNS change from the old cluster to a new cluster.</li></ol>

Scenario	Actions Required for Switchover
If you have multiple virtual hosts configured on your source system subscription, and want to retain those on your target system	<ol style="list-style-type: none"> <li>1. Create multiple virtual hosts on your target system. See <a href="#">Configuring Additional Virtual Host in Cloud Foundry Environment [page 124]</a>.</li> <li>2. Bind each API proxy to the desired virtual host on your target system.</li> </ol>

### Note

If your source and target belongs to the same data center and your source has a custom domain virtual host, and if you are planning to carry forward the same custom domain virtual host to target, please ensure that the following aspects are considered:

1. Since custom domain virtual host URL and port should be unique in a data center across tenants. It is not possible to have the same virtual host URL in both source and target at the same time. Therefore, delete the custom domain virtual host from source and then create the same custom domain virtual host in the target. To do this, you must create an incident on the component OPU-API-OD-OPS through the SAP Support Portal. For details, refer [Configuring Additional Virtual Host in Cloud Foundry Environment \[page 124\]](#).
2. When virtual host gets deleted in the source tenant, there will be downtime for all the APIs in the source account. The downtime will continue until the virtual host configuration gets completed. This configuration activity will require manual intervention by the API Management Operations team and also your DNS service provider for DNS cutover. We recommend that you plan this activity during your planned maintenance window.

### Switching Over Design time URLs of API portal and API business hub enterprise portals

- Domains managed by SAP can't be switched over.
- To switch over a custom domain, create an incident on the component OPU-API-OD-OPS through the [SAP Support Portal](#).

## 1.15.2.4 Recommendations

This topic lists the recommendations that you must consider for migration.

- After cloning the API Management content from the source to the target system for the first time, you must maintain both the systems, until you switch over from the source system to your target system completely.
- It is recommended that you always add or modify your entities in the source system, and clone it to the target system by rerunning the tool as and when required, instead of adding them directly to the target system.

## 1.15.2.5 Security Features of the Tenant Cloning Tool

The security features of the Tenant Cloning Tool is described in this section.

### Auditing and Logging

- The Tenant Cloning tool calls the APIs provided by and API business hub enterprise(developer portal). Hence, there are no security-related events available in the tool.
- All application logs generated from the cloning tool are stored in "APIM-Tenant-Cloning-Tool.log", an autogenerated log file.

### Data Protection and Privacy

- The tool doesn't persist any data on its own, nor is there a persistence layer.
- The tool logs the cloning status in the log file and in the output excel file named "apim-tct-output.xlsx".
  - The log and output excel file contain e-mail IDs of application developers, needed for troubleshooting and migration reporting, which are being cloned from source to target system.
  - The tool doesn't store any personal data (except e-mail IDs of application developers) in the log and output excel file.
  - We recommend storing the log file and output excel file securely, if further processing is needed; else these files must be deleted.
- The tool doesn't read any sensitive personal data.
- The tool doesn't change any personal data.

### Identity and Access Management

- No specific identity and access management configuration is needed to run the tool.
- Application developer's details are copied from source to target system as is. If some of the developer information isn't valid in the IDP configured in the target tenant, it must be corrected.

### Network and Communication Security

The tool uses standard HTTPS communication to make API calls, as provided by the and API business hub enterprise(developer portal).



## 2 Glossary

### Terms related to API Management

Entity	Description
API Management	<ul style="list-style-type: none"><li>• Creates simple digital experiences for your consumers, partners, and employees.</li><li>• Uses a technology that helps you to share digital assets and enable developer communities to consume these assets in new channels, devices, and user interfaces. Available in the cloud, the technology helps promote co- innovation among employees, partners, and the developer community.</li><li>• Reduces complexity by leveraging a single provisioning platform (API Platform) to provide unified access and governance of APIs across a heterogeneous landscape.</li><li>• Provides one experience for managing and monitoring all APIs across various data platforms and is enriched with real-time analytics and enables consumers to access relevant data directly in a secure manner. Selective data can be exposed while reducing the risk of security breaches.</li></ul>
SAP API Management	It lets you publish, promote, and oversee APIs in a secure and scalable environment
Neo environment	<p>SAP BTP, Neo environment contains SAP proprietary runtime. Neo is a feature-rich and easy-to-use development environment and allows you to develop Java, SAP HANA XS, and HTML5 applications.</p> <p>The Neo environment uses virtual machines, allowing you to install and maintain your own applications in scenarios that aren't covered by the platform.</p>
Cloud Foundry environment	<p>SAP BTP, Cloud Foundry environment contains the Cloud Foundry Application Runtime, which is based on the open-source application platform managed by the Cloud Foundry Foundation</p> <p>Application developers can use the Cloud Foundry environment to enhance SAP products and to integrate business applications, as well as to develop entirely new enterprise applications based on business APIs that are hosted on SAP BTP.</p>

Entity	Description
	The Cloud Foundry environment allows you to use multiple programming languages such as Java, Node.js, and community/bring-your-own language options.
<b>API Platform</b>	Provides tools to manage APIs and it facilitates the inclusion of new APIs, configuration of existing APIs, and helps you manage developers and apps. It also helps you create and consume APIs, whether you want to build API proxies as a service provider or use APIs, SDKs, and other convenient services as an app developer.
API Analytics	Provides powerful analytical tools to track your API usage. Use the API analytics to collect information on the IP, URL, user ID for API call information, latency data, and so on.
<b>Developer Services</b>	Provides tools to manage app developers. Provides the ability to onboard developers and creates a developer portal for publicly available products.
<b>API Management Account</b>	An API Management account is the highest level of data hierarchy. An account is a representation of all components including APIs, products, applications, systems, users, and developers.
<b>System</b>	In API Management System refers to the API provider systems where the actual backend services reside. The system could either be an ABAP system, SAP Gateway system, Enterprise Services Repository, or systems that host generic REST services or third-party provider systems. API Management allows you to add and manage an API provider system. After you have added a system, you can browse for the APIs in that system.
<b>User</b>	API Management can have multiple users. Different users have different roles and privileges assigned. For example, people who create APIs and products or analyze the metrics or the application consumer who can access the APIs provisioned by API Management.
<b>API</b>	APIs are Application Programming Interfaces. They comprise a set of routines, protocols, and tools for building software applications. APIs define sets of requirements that govern how applications communicate with one another. They facilitate interaction by selectively exposing certain functionalities, allowing different applications, websites, or devices to communicate effectively with each other

Entity	Description
Product	<p data-bbox="828 376 927 407">📌 Note</p> <p data-bbox="828 439 1337 495">API Management supports OData, REST, and SOAP services.</p> <p data-bbox="804 562 1385 719">A product is a bundle of APIs. It contains metadata specific to your business for monitoring or analytics. For example, all APIs related to CRM can be bundled as one CRM product. API Management collects data for analyzing the products.</p>
Developer	<p data-bbox="804 763 1385 853">One or more developers can create applications in the API Management account. A developer can consume APIs but cannot create APIs.</p> <p data-bbox="804 884 1385 1010">To create an application, the developer must have registered the account. After having created an application, the developer uses the app (application) key to consuming the APIs.</p>
Application	<p data-bbox="804 1048 1385 1279">Applications include the Web or mobile applications that consume the exposed APIs. When you create an application, you select the product to include in this application. For each application that you create, API Management generates an app key and secret. Use this key to gain access to multiple products. Developers create one or more applications using the APIs you expose.</p>
App Key	<p data-bbox="804 1317 1385 1514">Based on the authorization mechanism you define for your APIs; the application passes an app (application) key together with every request to your APIs. If that key is valid, the request is permitted. API Management supports different types of authentication, such as a simple API key, OAuth, and so on.</p>
API Portal	<p data-bbox="804 1547 1385 1615">You can browse through this API package for API Admin services with the required resources.</p>
Developer Portal	<p data-bbox="804 1653 1385 1720">You can browse through this API package for application development services that are offered.</p>
Metering	<p data-bbox="804 1753 1385 1843">You can now browse through this API package to view metering data for APIs, API Products, and applications in API Portal.</p>
API Analytics	

Entity	Description
<b>Client SDK</b>	<p>A client software development kit (SDK) is available for developers through a non-commercial license on open source sites.</p> <p>On the API Portal home page, choose the Client SDK. On selecting the client SDK, you are navigated to the maven repository, where you can download this package.</p>

## 3 Glossary-2

### Terms related to API Management

A-M

Entity	Description
Analyze APIs	Analyzing APIs helps you know more about API traffic data, and you can trace the API calls with real-time insights from your data.
API	API stands for Application Programming Interface. An API connects different software components by facilitating interaction with applications, websites, or devices to communicate effectively with each other by allowing data sharing.
API Analytics	API Analytics provides powerful analytical tools to track your API's health and usage. Using API Analytics, you get to utilize the sample analytical charts and key performance indicators (KPIs). These charts and KPIs are preconfigured in the dashboard.
API Call	An API call is a request made to the server using APIs.
API Gateway	An API Gateway is an execution engine that intercepts the API calls and executes the policies. The API Gateway adds on to the capabilities of API Management in security handling, traffic management, and governance.
API Design	An API design is used to provide an efficient interface and helps customers in a better understanding of the use-case of a product.
API Designer	An API Designer is a person who designs APIs. In the design phase, an API designer can define the requirements for APIs and plan the services (services like ODATA or REST) that can be used to expose to customers.
API Management Account	An API Management account is the highest level of data hierarchy. This account hosts the API platform, which is a combination of API and API business hub enterprise.
API Platform	API Platform enables enterprises to stimulate innovation, implement distributed services and data, adapts to market and customer needs. Hence, APIs have become the foundation of the fast-moving digital economy.

Entity	Description
API Portal	An API Portal provides tools to manage APIs, facilitates the inclusion of new APIs, and helps you manage them. It also helps you create APIs, whether you want to build API proxies as a service provider or use APIs, SDKs, and other convenient services as an app developer.
API Policy	API Management provides capabilities to define the behavior of an API by using 'policies.' A policy is a program that executes a specific function at runtime. They provide the flexibility to add common functionalities on an API without having to code them individually each time. Policies provide features to secure APIs, control the API traffic, and transform message formats. You can also customize the behavior of an API by adding scripts and attaching them to policies.
API Provider	An API provider is an abstraction for a system that defines the connection details for services running on specific hosts whose details you want to access. It also displays data through a programmatically consumable service or an API.
API Proxy	An API proxy is a masked URL that disconnects the app-surfacing API from your backend services, protecting those apps from backend code changes.
Application	Applications include the Web or mobile applications that consume the exposed APIs. When you create an application, you select the product to include in this application. For each application that you create, API Management generates an app key and secret. Use this key to gain access to multiple products.
Application Key	An application key is an encryption code assigned to a specific application. Different applications have different application keys. This key validates if the API requested was associated to a particular application. After having created an application, the developer uses the application key to consume the APIs.
Build APIs	Build API proxies prominently and configure API policies as steps in the API flow. Customize API behavior using code. Plus, modify from or to any protocol.
Client SDK	<p data-bbox="804 1756 1390 1852">A client software development kit (SDK) is available for developers through a noncommercial license on open-source sites.</p> <p data-bbox="804 1868 1390 1973">On the API Portal home page, choose the Client SDK. On selecting the client SDK, you are navigated to the maven repository, where you can download this package.</p>

<b>Entity</b>	<b>Description</b>
Consume API	Consume or use APIs via the API business hub enterprise. In the API business hub enterprise, an application developer registers, explores the API exposed by customers, creates applications, and tests APIs.
Cloud Foundry Environment	SAP BTP Cloud Foundry environment contains the Cloud Foundry Application Runtime, which is based on the open-source application platform managed by the Cloud Foundry Foundation.
API business hub enterprise	A personalized portal that lets you to instantly explore, test, get API keys, and innovate quick. You can also accelerate API adoption with offerings, rate limits, and pricing.
Developer Services	Services that provide tools to manage app developers. Also, provide the ability to onboard developers and creates a API business hub enterprise for publicly available products.
Keystore	A keystore contains the SSL Certificate and private key that is used to identify the entity during SSL Handshake.
Monetize APIs	Allows the API Providers to generate revenue via a rate plan service or billing service for the API usage.
N-Z	
<b>Entity</b>	<b>Description</b>
NEO Environment	SAP BTP NEO environment contains SAP propriety runtime. NEO is a feature-rich and easy-to-use development environment, allowing you to develop Java, SAP HANA XS, and HTML5 applications.
Policy Template	A policy template is a policy structure designed to define the behavior of an API policy. You can create, apply, update, import, export, and delete a policy template using the API Portal.
Publish API	Publish the API product on the API portal. Make it available for consumption.
Product	A product is a package that contains APIs and metadata specific to your business for monitoring or analytics. For example, all APIs related to CRM can be bundled as one CRM product.
Rate Plan	Rate Plan is the price per API call made to any external app service. You can decide the basic charge, mode of currency, and rate plan type while creating a rate plan for all the API calls made.

Entity	Description
SAP API Management	A service that offers you a harmonized experience for creating, maintaining, governing, and monitoring all your APIs across data-platforms. You can also leverage real-time analytics to make quick business decisions that are critical in today's API-first strategy.
System	<p>In API Management, a system refers to the API provider system where the actual backend services reside. The system could either be an ABAP system, SAP Gateway system, Enterprise Services Repository, or systems that host generic REST services or third-party provider systems.</p> <p>For example, API Management allows you to add and manage an API provider system. After you have added a system, you can browse for the APIs in that system.</p>
Test Console	API Management provides an API Test Console, which enables you to test your APIs. Testing an API is essential to understand the runtime behavior of the APIs. The test console allows you to explore the resources associated with an API and execute the operations.
Trust Store	A truststore contains certificates used to validate certificates obtained as part of SSL handshaking.
User	API Management can have multiple users. Different users have different roles and privileges assigned to them. For example, people who create APIs and products or analyze the metrics.



# 4 Reuse Content for Cloud Foundry Plans

## Deleting an API Management Service Instance


Delete an API Management service instance.

### Prerequisites

- You have the space developer role assigned to you.
- You have created an API Management service instance.

Use the following procedure to delete an API Management service instance on Cloud Foundry.

### Procedure

1. In your Web browser, open the SAP BTP Cockpit.
2. In the provider account, choose [Services](#) > [Service Marketplace](#) > [Instances](#) >
3. Select the API Management service tile.
4. From the list of instances visible, select the instance that you want to delete and choose .
5. Choose *OK*.

## Updating an API Management Service Instance

Update user credentials for an API Management service instance.

### Prerequisites

- You have created an API Management service instance.
- You have logged on as a space developer.

### Context

Perform the following steps to update user credentials for an API Management service instance.

### Procedure

Open the command-line interface for Cloud Foundry and enter the following command:

#### Note

You can update a service only from the command-line interface and not from SAP Cloud BTP cockpit.

#### Sample Code

```
cf update-service apim-service-instance-name -c '{"apiportal_admin" : "<api portal admin id>", "apiportal_password" : "<api portal password>", "consent" : <value should be true>}'
<-- Example
```

```
// For Linux/MAC system
cf update-service apim-prod-instance -c '{"apiportal_admin" : "USER",
"apiportal_password" : "PASSWORD, "consent" : true}'
// For Windows system
cf update-service instance102 -c "{\"apiportal_admin\": \"user\",
\"apiportal_password\": \"password\", \"consent\" : true}"
-->
```

### Sample Code

For more information on Updating a service, see [Update Service](#).

## Binding a Cloud Foundry Application to an API Management Service Instance

Create a service instance and bind the Cloud Foundry application to API management. When you bind an application, an API proxy is created and a new route is added to the application. The route initially redirects all calls to the proxy URL and then to the application.

### Prerequisites

- You have created an API Management service instance.
- You have logged on as a space developer.

Open the command-line interface for Cloud Foundry and enter the following command:

### Sample Code

```
cf bind-route-service sap-cf-domain.com apim-service-instance-name --hostname
my-app -c '{"api_name" : "custom_api_proxy_name"}'
<-- Example
//Without parameters
cf bind-route-service cfapps.sap.hana.ondemand.com apim-prod-instance --
hostname taxapp
//Cloud foundry URL for the above example is https://
taxapp.cfapps.sap.hana.ondemand.com
//With parameters for Linux/MAC system
cf bind-route-service sap-cf-domain.com apim-service-instance-name --hostname
my-app -c '{"api_name" : "test_api"}'
//With parameters for Windows system
cf bind-route-service sap-cf-domain.com apim-service-instance-name --hostname
my-app -c "{\"api_name\" : \"test_api\"}"
-->
```

### Note

API Management supports only English alpha numeric, hyphens (-) and underscores (\_) characters for "api\_name".

You can bind an application to a service only from the command-line interface and not from SAP BTP Cockpit.

Providing a value for the parameter during binding is optional. If you provide a value for api\_name, then the API proxy created in API portal for current binding gets the given name. Also, if an API with the same name

exist in the API portal, then the same API proxy is used for the binding. That is, the API proxy end point is registered as the route service URL for the current binding.

For more information on binding an application, see [bind route service](#) .

## Unbinding a Cloud Foundry Application from an API Management Service Instance

Unbind an application to an API Management by deleting the service instance. When you unbind an application, API proxy is eliminated from the application.

### Prerequisites

- You have logged on as a space developer
- You have bound an application to an API Management service instance.

Open the command prompt and enter the following command:

### Sample Code

```
cf unbind-route-service sap-cf-domain.com apim-service-instance-name --
hostname my-app
<-- Example
cf unbind-route-service cfapps.sap.hana.ondemand.com apim-prod-instance --
hostname taxapp
-->
```

### Note

You can unbind an application from a service only from the command-line interface and not from SAP BTP Cockpit.



For more information on unbinding an application, see [Unbind route service](#) .

# Important Disclaimers and Legal Information

## Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.

About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

## Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

## Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.

The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

## Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

## Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.



© 2024 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see <https://www.sap.com/about/legal/trademark.html> for additional trademark information and notices.