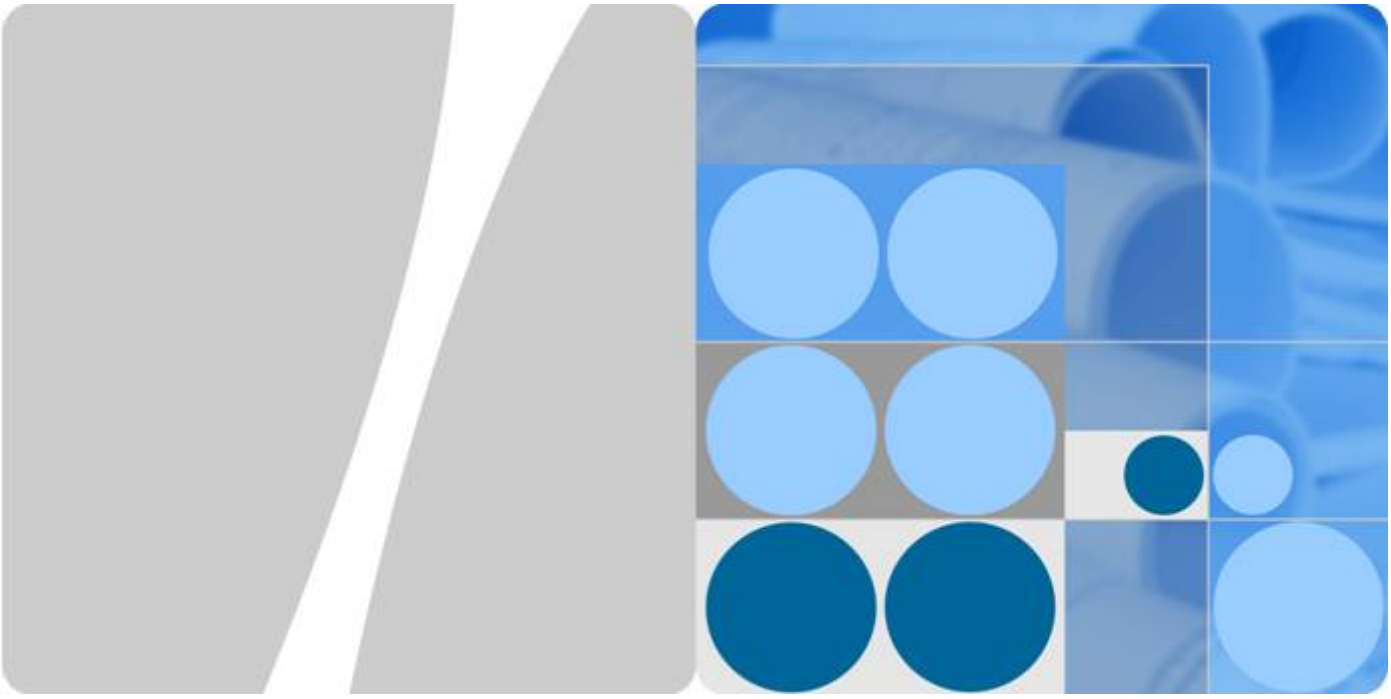


API Gateway

User Guide

Date **2021-10-25**

Contents

1 Overview

API Gateway is a fully managed service that enables you to securely build, manage, and deploy APIs at any scale with high performance and availability. With API Gateway, you can easily integrate your internal service systems and selectively expose and monetize your APIs.

This document provides guidance to enterprises and developers on selectively exposing their services and data through API Gateway and monetizing their APIs. It also describes how to obtain and call APIs of other providers to reduce development time and costs.

2 Getting Started

2.1 Introduction

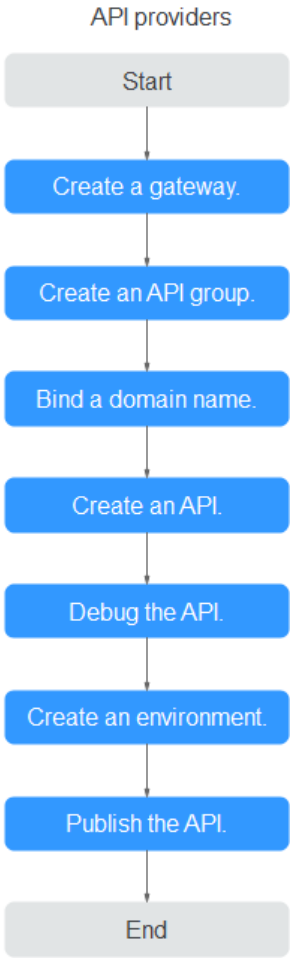
API Gateway is a fully managed service that enables you to securely build, manage, and deploy APIs at any scale with high performance and availability. With API Gateway, you can easily integrate your internal service systems and selectively expose and monetize your APIs.

To learn about the process of exposing and calling an API, see **Opening APIs** and **Calling APIs**. **Simple authentication** with an app is used for illustration.

2.2 Opening APIs

2.2.1 Process Flow

The following figure shows the process of exposing an API.



1. Creating a Gateway

   Use the shared gateway.

2. **Creating an API Group**

   An API group facilitates management of APIs used for the same service. Create an API group and then create APIs.

3. **Binding a Domain Name**

   Before making the API available for users to access, bind an independent domain name (custom domain name) to the group to which the API belongs. Then API callers can use these domain names to call the API.

4. **Creating an API**

   When creating an API, configure the frontend and backend request paths, parameters, and protocols.

5. **Debugging an API**

   Debug the API to check whether it works normally.

6. **(Optional) Creating an Environment**

   An API can be called in different scenarios, such as the production environment (RELEASE) or other custom environments. RELEASE is the default environment defined in API Gateway.

7. **Publishing an API**

   Publish the API so that it can be called.

2.2.2 Creating an API Group

Step 1 Log in to the API Gateway console.

Step 2 In the navigation pane, choose **API Publishing** > **API Groups**.

Step 3 Click **Create API Group** and configure the group information.

Table 2-1 API group information

Parameter	Description
Name	API group name. It is recommended that you enter a name based on naming rules to facilitate search.
Description	Description of the API group.

Step 4 Click **OK**. The system automatically allocates a subdomain name to the API group. APIs in the group can be debugged using the subdomain name.

----End

2.2.3 Binding a Domain Name

Step 1 On the **API Groups** page, click the group created in **Creating an API Group** to go to the group details page.

Step 2 Click the **Domain Names** tab.

Step 3 Click **Bind Independent Domain Name** and enter a domain name.

NOTE

The independent domain name must have been registered and CNAMEd to the subdomain name of the API group.

Step 4 Click **OK**.

----End

2.2.4 Creating an API

Create an API with the following steps:

1. **Setting Basic Information**
2. **Defining API Request**
3. **Defining Backend Service**
4. **Defining Responses**

Setting Basic Information

Step 1 In the navigation pane, choose **API Publishing** > **APIs**.

Step 2 Click **Create API** and set the basic information.

Table 2-2 Setting basic information

Parameter	Description
Name	API name. It is recommended that you enter a name based on naming rules to facilitate search.
API Group	By default, the group created in **Creating an API Group** is selected.
Gateway Response	Select a response to be displayed if API Gateway fails to process an API request. The default gateway response is **default**.
Visibility	By default, **Public** is selected.
Security Authentication	API authentication mode. Set this parameter to **App**.
Simple Authentication	If you enable this option, API Gateway verifies only the AppCode and the request signature does not need to be verified. For this example, enable simple authentication.
Tag Name	Classification attribute used to quickly identify the API from other APIs.
Description	Description of the API.

Step 3 Click **Next**.

----End

Defining API Request

Step 1 On the **Define API Request** page, set the request information.

Table 2-3 Parameters for defining API requests

Parameter	Description
Domain Name	The subdomain automatically allocated to the API group created in **Creating an API Group**.
Protocol	Request protocol of the API. Set this parameter to **HTTPS**.
Path	The path for requesting the API.
Matching	By default, **Exact match** is selected.
Method	Request method of the API. Set this parameter to **POST**.
CORS	For this example, disable this option.

Step 2 Click **Next**.

----End

Defining Backend Service

Step 1 On the **Define Backend Request** page, set the backend service information.

Step 2 Select a backend service type. For this example, select **HTTP/HTTPS**.

Table 2-4 Parameters for defining an HTTP/HTTPS backend service

Parameter	Description
Protocol	Set this parameter to **HTTP**.
Method	Request method of the API. Set this parameter to **POST**.
VPC Channel	Determine whether the backend service will be accessed using a VPC channel. For this example, select **Skip**.
Backend Address	Address of the backend service.
Path	Path of the backend service.
Timeout	Backend service request timeout. Default value: 5000 ms.
Backend Authentication	Determine whether your backend service needs to authenticate API requests. For this example, disable this option.

Step 3 Click **Next**.

----End

Defining Responses

Step 1 On the **Define Response** page, set the responses.

Table 2-5 Defining responses

Parameter	Description
Example Success Response	An example of a response returned when the API is called successfully.
Example Failure Response	An example of a response returned when the API fails to be called.

Step 2 Click **Finish**.

----End

2.2.5 Debugging an API

Step 1 On the **APIs** page, locate the API created in **Creating an API**, and choose **More** > **Debug**.

Step 2 On the left side, set the API request parameters listed in **Table 2-6**. On the right side, view the API request and response information after you click **Send Request**.

Table 2-6 Parameters for debugging an API

Parameter	Description
Protocol	This parameter can be modified only if you set **Protocol** to **HTTP&HTTPS** for the API.
Method	This parameter can be modified only if you set **Method** to **ANY** for the API.
Path	Request path of the API.
Query Strings	Query string parameters and values.
Headers	HTTP headers and values.
Body	This parameter can be modified only if you set **Method** to **PATCH**, **POST**, or **PUT** for the API.

Step 3 Click **Send Request**.

If the API is called successfully, the status code **200** is displayed.

----End

2.2.6 (Optional) Creating an Environment

Step 1 In the navigation pane, choose **API Publishing** > **Environments**.

Step 2 Click **Create Environment** and set the environment information.

Table 2-7 Environment information

Parameter	Description
Name	Environment name. It is recommended that you enter a name based on naming rules to facilitate search.
Description	Description of the environment.

Step 3 Click **OK**.

----End

2.2.7 Publishing an API

Step 1 In the navigation pane, choose **API Publishing** > **APIs**.

Step 2 Locate the API created in **Creating an API**, and click **Publish**.

Step 3 Select the environment where the API will be published.

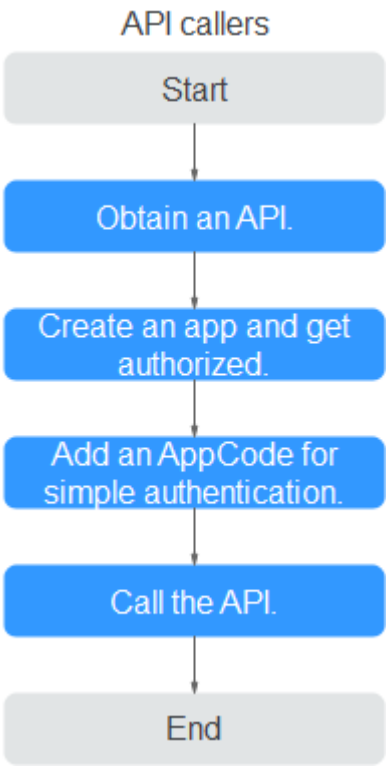
Step 4 Click **Publish**.

----End

2.3 Calling APIs

2.3.1 Process Flow

The following figure shows the process of calling an API.



1. Obtaining an API

   Obtain an API and its documentation from an API provider.

2. **Creating an App and Getting Authorized**

   APIs that use app authentication can only be called using apps bound to them.

3. **Adding an AppCode for Simple Authentication**

   API Gateway only verifies the AppCode during simple authentication.

4. **Calling the API**
   Use an API test tool to call the API with app authentication credentials.

2.3.2 Creating an App and Getting Authorized

Creating an App

Step 1 In the navigation pane, choose **API Calling** > **Apps**.

Step 2 Click **Create App** and set basic app information.

Table 2-8 App information

Parameter	Description
Name	App name. It is recommended that you enter a name based on naming rules to facilitate search.
Description	Description of the app.

Step 3 Click **OK**.

Step 4 Click the app, and view the AppKey and AppSecret on the app details page.

----End

Binding an App to an API

Step 1 At the top of the API list, click **Select API**.

Step 2 Select the environment, API group, and API created in **Opening APIs**, and click **OK**.

----End

2.3.3 Adding an AppCode for Simple Authentication

Step 1 In the app list, click the app created in **Creating an App and Getting Authorized** to go to the app details page.

Step 2 Click the **AppCodes** tab.

Step 3 Click **Add AppCode**.

Step 4 Select **Automatically generated**.

Step 5 Click **OK**.

----End

2.3.4 Calling an API

Use an API test tool to configure the API calling information.

Step 1 Obtain the API request information.

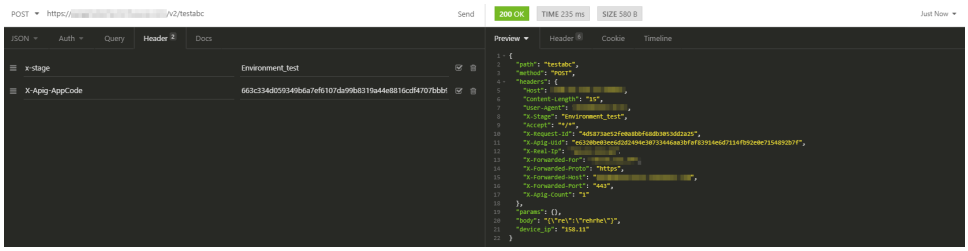
For illustration purposes, an API and its documentation are obtained through offline channels. You can also obtain the authentication mode, request method, request path, and other information about the API.

Step 2 Add the header parameter **X-Apig-AppCode** and set the parameter value to the **generated AppCode**.

Step 3 Add the header parameter **x-stage** and set the parameter value to the **running environment**. Skip this step if the API has been published in the RELEASE environment.

Step 4 Click **Send** to send a request.

If the API is called successfully, the message **200 OK** is displayed.



----End

3 API Opening

3.1 API Group Management

3.1.1 Creating an API Group

Scenario

Before creating an API, you must create an API group. An API group contains different APIs used for the same service.

NOTE

- Each API can only belong to one API group.
- You can create a maximum of 50 API groups.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **API Groups**.

Step 4 Click **Create API Group**, and set the parameters described in **Table 3-1**.

Table 3-1 Parameters for creating an API group

Parameter	Description
Name	API group name.
Description	Description of the API group.

Step 5 Click **OK**.

After the API group is created, it is displayed in the API group list.

NOTE

- The system automatically allocates a subdomain name to the API group for internal testing. The subdomain name can be accessed 1000 times a day.
- To make your APIs available for users to access, bind independent domain names to the API group to which the APIs belong.

----End

Follow-Up Operations

After the API group is created, bind independent domain names to it so that API callers can use the domain names to call APIs in the group. For more information, see **Binding a Domain Name**.

3.1.2 Binding a Domain Name

Scenario

Before you open an API, you must bind one or more independent domain names to the group to which the API belongs. If no domain names are bound to the group, the API will be called using the default subdomain name of the group and can be called only 1000 times a day.

NOTE

You can bind a maximum of five independent domain names to an API group.

Note the following points before you bind a domain name:

- Subdomain name: After an API group is created, the system automatically allocates a unique subdomain name to it for internal testing. The subdomain name can be accessed 1000 times a day, but it cannot be modified.
- Independent domain name: An independent domain name is a custom domain name used for API callers to call open APIs in the group to which the domain name is bound.

Prerequisites

1. There is an independent domain name available.
2. A CNAME record points the independent domain name to the subdomain name of the API group. For details, see chapter "Managing Record Sets" in the *Domain Name Service User Guide*.
3. If the API group contains APIs that are called through HTTPS, there needs to be SSL certificates configured for the independent domain name. SSL certificates can only be added manually with a custom name, content, and a key.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **API Groups**.

Step 4 Go to the **Domain Names** tab page using one of the following methods:

- Click the name of the target API group, and click the **Domain Names** tab on the displayed API group details page.
- In the **Operation** column of the target API group, choose **More** > **Manage Domain Name**.

Step 5 Click **Bind Independent Domain Name** and enter a domain name.

Step 6 Click **OK**.

If the domain name is not needed, click **Unbind** to unbind it from the API group.

Step 7 (Optional) If the API group contains APIs that are accessed through HTTPS, add an SSL certificate.

1. Click **Add SSL Certificate**.
2. Enter the name, content, and key of the **obtained SSL certificate**, and click **OK**.

NOTE

- Currently, you can only add SSL certificates in the PEM format. To add SSL certificates of other formats, convert the certificates into the PEM format first.
- To edit an SSL certificate, click  next to the certificate name. The certificate content and key will not be visible after you click **OK** to add the certificate. If the content has been updated, add the entire content or key again.
- If you do not require an SSL certificate, click **Delete SSL Certificate** in the row containing the certificate to delete it.

----End

Troubleshooting

- Failure in binding an independent domain name: The independent domain name is not CNAMEd to the subdomain name of the API group, or the independent domain name already exists.

- Failure in adding an SSL certificate: The domain name of the SSL certificate is different from the domain name for which you add the SSL certificate.

Follow-Up Operations

After binding independent domain names to the API group, create APIs in the group to selectively expose backend capabilities. For details, see **Creating an API**.

3.1.3 Deleting an API Group

Scenario

You can delete an API group if you do not require it.

Prerequisites

You have created an API group.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **API Groups**.

Step 4 Delete an API group. You can use one of the following methods:

- In the **Operation** column of the target API group, choose **More** > **Delete**.
- Click the name of the target API group, and click **Delete Group** in the upper right corner of the displayed API group details page.

Step 5 Enter **DELETE** and click **Yes**.

----End

3.1.4 Adding a Gateway Response

Scenario

A gateway response is displayed if API Gateway fails to process an API request. API Gateway provides a set of default responses and also allows you to create gateway responses with custom status codes and content, on the **API Groups** page. The response content must be in JSON format.

For example, the content of a default gateway response is as follows:

```
{"error_code": "$context.error.code", "error_msg": "$context.error.message", "request_id": "$context.requestId"}
```

You can add a response with the following content:

```
{"errorcode": "$context.error.code", "errormsg": "$context.error.message", "requestid": "$context.requestId","apiId":"$context.apiId"}
```

You can add more fields to or delete existing fields from the JSON body.

NOTE

- The default gateway responses provided by API Gateway can be edited.
- You can create gateway responses and configure different responses for APIs in the same API group.
- The type of a gateway response cannot be changed. For details, see **Response Types**.
- Gateway responses can contain the API gateway context variables (starting with **$context**). For details, see **API Gateway Context Variables**.

Prerequisites

You have created an API group.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **API Groups**.

Step 4 Locate the API group for which you want to create or modify a gateway response, and click the group name to go to the API group details page.

Step 5 Click the **Gateway Responses** tab and create a gateway response.

NOTE

- To edit a response, click the **Edit** button in the upper right corner and modify the status code and content of the response.
- You can modify only the status code and content of a default or custom gateway response, and you cannot change the response type.
- Error information and other response details can be obtained using variables. For details about the supported variables, see **Table 3-3**.

----End

Response Types

Table 3-2 lists the response types supported by API Gateway. You can define status codes of responses to meet your service requirements.

Table 3-2 Error Response types supported by API Gateway

Response Name	Default Status Code	Description
Access Denied	403	Access denied. For example, the access control policy is triggered or an attack is detected.

Response Name	Default Status Code	Description
Authorizer Configuration Error	500	A custom authorizer error has occurred. For example, communication failed or an error response was returned.
Authorizer Failed	500	The custom authorization failed.
Incorrect Identity Source	401	The identity source of the custom authorizer is missing or invalid.
Authentication Failure	401	IAM or app authentication failed.
Identity Source Not Found	401	No identity source has been specified.
Backend Timeout	504	Communication with the backend service timed out.
Backend Unavailable	502	The backend service is unavailable due to communication error.
Default 4XX	-	Another 4XX error occurred.
Default 5XX	-	Another 5XX error occurred.
No API Found	-	No API is found.
Incorrect Request Parameters	404	The request parameters are incorrect or the HTTP method is not supported.
Request Throttled	429	The request was rejected due to request throttling.
Unauthorized App	401	The app you are using has not been authorized to call the API.

API Gateway Context Variables

Table 3-3 Variables that can be used in response message body

Variable	Description
$context.apiId	API ID.
$context.appId	ID of the app that calls the API.
$context.requestId	Tracing ID generated when the API is called.
$context.stage	Deployment environment in which the API is called.

Variable	Description
$context.sourceIp	Source IP address of the API caller.
$context.authorizer.frontend.property	Values of the specified attribute–value pairs mapped to the context in the frontend custom authorizer response
$context.authorizer.backend.property	Values of the specified attribute–value pairs mapped to the context in the backend custom authorizer response
$context.error.message	Error message.
$context.error.code	Error code.
$context.error.type	Error type.

3.2 API Management

3.2.1 Creating an API

Scenario

You can selectively expose your services by configuring their APIs in API Gateway.

To create an API, set the basic information and define the API request, backend service, and responses.

NOTE

- API Gateway uses a REST-based API architecture, so API opening and calling must comply with related RESTful API specifications.
- You can create a maximum of 200 APIs.

Prerequisites

- You have created an API group. If no API group is available, create one during API creation.
- If the backend service of the API is deployed in a VPC, you have created a VPC channel to access the service by following the procedure in **Creating a VPC Channel**. You can also create a VPC channel during API creation.

Setting Basic Information

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **APIs**.

Step 4 Click **Create API**, and set the parameters listed in **Table 3-4**.

Table 3-4 Basic information

Parameter	Description
Name	API name. It is recommended that you enter a name based on naming rules to facilitate search.
API Group	The group to which the API belongs. If no API group is available, click **Create API Group** to create one.
Gateway Response	Displayed if API Gateway fails to process an API request. API Gateway provides a set of default responses and also allows you to **create gateway responses** with custom status codes and content, on the **API Groups** page. The response content must be in JSON format.
Visibility	Determine whether the API is available to the public. Options: • **Public**
Security Authentication	The following authentication modes are available: • **App**: Requests for the API will be authenticated by API Gateway. • **IAM**: Requests for the API will be authenticated by Identity and Access Management (IAM). • **Custom**: Requests for the API will be authenticated by using your own authentication system or service (for example, an OAuth-based authentication system). • **None**: No authentication will be required. The API calling method varies depending on the authentication mode. For details, see *Developer Guide*. App authentication is recommended. **NOTICE** • If you set the authentication mode of an API to **IAM**, any API Gateway user can access the API, which can result in excessive charges if the API is bombarded with malicious requests. • If you set the authentication mode of an API to **None**, any user can access the API over public networks, which can result in excessive charges if the API is bombarded with malicious requests. • If you set the authentication mode of an API to **Custom**, you can create a function in FunctionGraph to interconnect with your own authentication system or service. This authentication mode is not supported in regions where FunctionGraph is unavailable.

Parameter	Description
Simple Authentication	This parameter is available only if you set **Security Authentication** to **App**. If you select app authentication, you can configure whether to enable simple authentication. In simple authentication, the **X-Apig-AppCode** parameter is added to the HTTP request header for quick response. API Gateway verifies only the AppCode and the request content does not need to be signed. Simple authentication only supports HTTPS requests and does not support HTTP requests. For details, see **Adding an AppCode for Simple Authentication**. **NOTE** After you enable simple authentication for an existing API, you need to publish the API again. For details, see **Publishing an API**.
Custom Authorizer	This parameter is mandatory if **Security Authentication** is set to **Custom**. Select a custom authorizer if you set **Security Authentication** to **Custom**. If no custom authorizer is available, click **Create Custom Authorizer** to create one.
Tag Name	Classification attribute used to quickly identify the API from other APIs.
Description	Description of the API.

Step 5 Click **Next**.

----End

Defining API Request

Step 1 On the **Define API Request** page, set the parameters listed in **Table 3-5**.

Table 3-5 Parameters for defining API requests

Parameter	Description
Domain Name	The subdomain automatically allocated to the API group.
Protocol	The protocol used for calling the API. Options: • HTTP • HTTPS • HTTP&HTTPS HTTPS is recommended for transmitting important or sensitive data.

Parameter	Description
Path	The path for requesting the API. Enter a path in the format of "/users/*{userId}*/projects". ● The variable in braces ({}) is a request parameter. Ensure that it is an entire segment between a pair of slashes (/). A segment that is not marked by a pair of slashes, for example, **/abc{userId}**, is not supported. If you set the matching mode to **Exact match**, you can add a plus sign (+) to the end of the request parameter, for example, **/users/{*p*+}**. The variable *p* matches the segments between one or multiple pairs of slashes (/). ● Ensure that you define the parameters contained in the request path as input parameters. ● The content is case-sensitive.
Matching	Options: ● **Exact match**: The API can be called only using the specified request path. ● **Prefix match**: The API can be called using paths starting with the matching characters. For example, if you set the request path to **/test/AA** and the matching mode to **Prefix match**, the API can be called using **/test/AA/CC** but cannot be called using **/test/AACC**. **NOTE** If you set the matching mode to **Prefix match**, the characters of the API request path excluding the prefix are transparently transmitted to the backend service. For example, if you define the frontend and backend request paths of an API as **/test/** and **/test2/**, respectively, and the API is called using **/test/AA/CC**, the characters **AA/CC** will be transparently transmitted to the backend service. The request URL received by the backend service is **/test2/AA/CC/**.
Method	The API calling method. The options are **GET**, **POST**, **DELETE**, **PUT**, **PATCH**, **HEAD**, **OPTIONS**, and **ANY**. ● **ANY** indicates that the API can be called using any request method. ● If you set **Method** to **POST**, **PUT**, **PATCH**, or **ANY**, set the request body.

Parameter	Description
CORS	Determine whether to enable cross-origin resource sharing (CORS). CORS allows browsers to send XMLHttpRequest to servers in other domains, overcoming the limitation that Asynchronous JavaScript and XML (AJAX) can be used only within the same domain. There are two types of CORS requests: • Simple requests: requests that have the **Origin** field in the header. • Not-so-simple requests: HTTP requests sent before the actual request. If you enable CORS, you need to create another API that uses the OPTIONS method. For details, see **CORS**.

Step 2 (Optional) Set input parameters.

Input parameters are transmitted together with the request when the API is called.

1. Click **Add Input Parameter**.
2. Set the parameters listed in **Table 3-6**.

Table 3-6 Input parameter definition

Parameter	Description
Name	Name of the input parameter. If you set the parameter location to **PATH**, ensure that the parameter name is the same as that defined in the request path. **NOTE** – The parameter name is not case-sensitive. It cannot start with **x-apig-** or **x-sdk-**. – The parameter name cannot be **x-stage**. – If you set the parameter location to **HEADER**, ensure that the parameter name is not **Authorization** or **X-Auth-Token** and does not contain underscores (_).
Location	Position of the parameter in requests. The options are **PATH**, **HEADER**, and **QUERY**. **NOTE** If you set the parameter location to **PATH**, you must include the parameter in the request path.
Type	Type of the parameter value. Options: **STRING** and **NUMBER**. **NOTE** If the input parameter is of the Boolean type, select **String**.

Parameter	Description
Mandatory	Determine whether the input parameter is required in each request sent to call the API. If you select **Yes**, API requests that do not contain the input parameter will be rejected.
Default Value	The value that will be used if no value is specified for the input parameter when the API is called. If the input parameter is not specified in a request, API Gateway will automatically send the default value to the backend service.
Minimum Length	The minimum length of the parameter value. Only numbers are allowed.
Maximum Length	The maximum length of the parameter value. Only numbers are allowed.
Example	Example value for the parameter.
Description	Description of the parameter.

3. Click **OK**.

Step 3 Click **Next**.

----End

Defining Backend Service

API Gateway allows you to define multiple backend policies for different scenarios. Requests that meet specified conditions will be forwarded to the corresponding backend. For example, you can have certain requests to an API forwarded to a specific backend by specifying the source IP address in the policy conditions of the backend.

You can define a maximum of five backend policies for an API in addition to the default backend.

Step 1 Define the default backend.

API requests that do not meet the conditions of any backend will be forwarded to the default backend.

On the **Define Backend Request** page, select a backend type.

Table 3-7, **Table 3-8**, and **Table 3-9** describe the backend service parameters.

Table 3-7 Parameters for defining an HTTP/HTTPS backend service

Parameter	Description
Protocol	HTTP or HTTPS. This protocol must be the one used by the backend service. **NOTE** • WebSocket is supported for HTTP and HTTPS. • HTTPS is recommended for transmitting important or sensitive data.
Method	The API calling method. The options are **GET**, **POST**, **DELETE**, **PUT**, **PATCH**, **HEAD**, **OPTIONS**, and **ANY**. **ANY** indicates that the API can be called using any request method.
VPC Channel	Determine whether the backend service will be accessed using a VPC channel. • **If yes, select a VPC channel.** **NOTE** – To ensure a successful health check and service availability, configure the security groups of cloud servers in each VPC channel to allow access from 100.125.0.0/16. • **If no, configure the backend service address.** Enter a backend address in the format of "host IP address or domain name":"port number". The default port (80 for HTTP and 443 for HTTPS) will be used if you do not specify a port. Available ports: 1 to 65535. If you want to use a variable, enclose the variable name in number signs (#), for example, **#ipaddress#**. You can use multiple variables, for example, **#ipaddress##test#**.
Host Header (if applicable)	This parameter is available only if you set **VPC Channel** to **Configure**. Define a host header for requests to be sent to cloud servers associated with the VPC channel. By default, the original host header in each request will be used.
Path	The request path (URI) of the backend service. Ensure that any parameters in the path are enclosed in braces ({}). For example, **/getUserInfo/{userId}**. If the path contains an environment variable, enclose the environment variable in number signs (#), for example, **/#path#**. You can use multiple environment variables, for example, **/#path##request#**.
Timeout (ms)	Backend request timeout. If a backend timeout error occurs during API debugging, increase the timeout to locate the reason.

Parameter	Description
Backend Authentication	Determine whether your backend service needs to authenticate API requests. If you enable this option, select a custom authorizer for backend authentication. **Custom authorizers** are functions that are created in FunctionGraph to implement an authentication logic or to invoke an authentication service. **NOTE** Backend authentication relies on FunctionGraph and is only available in certain regions.

Table 3-8 Parameters for defining a FunctionGraph backend service

Parameter	Description
FunctionURN	Identifier of the requested function. Click **Select Function URN** to specify a function URN.
Version	Version of the function.
Invocation Mode	• **Synchronous**: synchronous invocation. When receiving an invocation request, FunctionGraph immediately processes the request and returns a result. The client closes the connection once it has received a response from the backend. • **Asynchronous**: asynchronous invocation. The function invocation results of client requests do not matter to clients. When it receives a request, FunctionGraph queues the request, returns a response, and then processes requests one by one in idle state.
Timeout (ms)	Backend request timeout. For details, see **Table 3-7**.
Backend Authentication	For details, see the description about backend authentication in **Table 3-7**.

Table 3-9 Parameters for defining a Mock backend service

Parameter	Description
Response	You can use Mock for API development, debugging, and verification. It enables API Gateway to return a response without sending the request to the backend. This is useful if you need to test APIs when the backend is unavailable.
Backend Authentication	For details, see the description about backend authentication in **Table 3-7**.

NOTE

- If you have defined an environment variable in the backend request path, the API cannot be debugged on the API debugging page.
- For variables defined in the backend request path of an API, corresponding environment variables and their values must be configured. Otherwise, the API cannot be published because there will be no values that can be assigned to the variables.
- Environment variable names are case-sensitive.

Step 2 (Optional) Add a backend policy.

You can add backend policies to forward requests to different backend services.

1. Click **Add Backend Policy**.
2. Set parameters by referring to **Table 3-10** and **Table 3-7**.

Table 3-10 Backend policy parameters

Parameter	Description
Name	The backend policy name.
Effective Mode	– **Any condition met**: The backend policy takes effect if any of the policy conditions has been met. – **All conditions met**: The backend policy takes effect only when all the policy conditions have been met.
Policy Conditions	Conditions that must be met for the backend policy to take effect. Set conditions by referring to **Table 3-11**.

Table 3-11 Policy conditions

Parameter	Description
Source	– Source IP address – Input parameter **NOTICE** Input parameters (for example, headers) set as policy conditions must have already been defined in the API request settings.
Parameter Name	– When setting **Source** to **Input parameter**, select an input parameter.
Parameter Location	The parameter location is displayed only if you set **Source** to **Input parameter**.

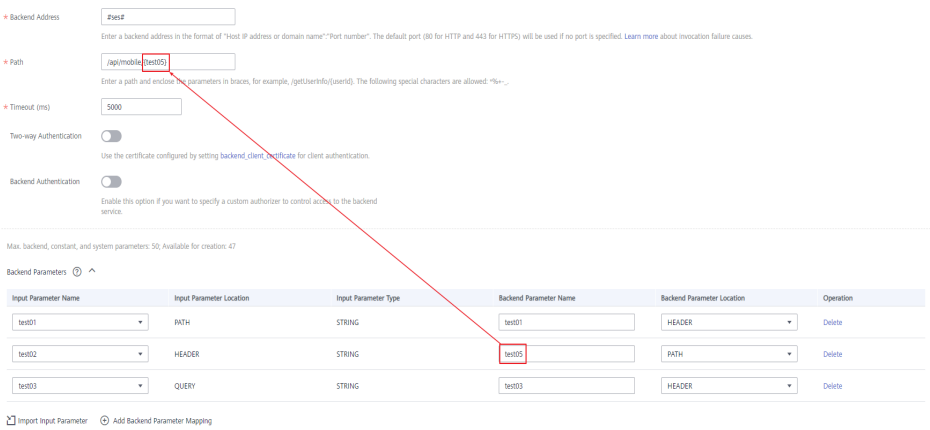
Parameter	Description
Condition Type	This parameter is required only if you set **Source** to **Input parameter**. – **Equal**: The request parameter must be equal to the specified value. – **Enumerated**: The request parameter must be equal to any of the enumerated values. – **Matching**: The request parameter must be equal to any value of the regular expression.
Condition Value	Set a condition value according to the condition type. – **Equal**: Enter a value. – **Enumerated**: Enter multiple values and separate them using commas. – **Matching**: Enter a range, for example, **[0-5]**. If you have set **Source** to **Source IP address**, enter one or more IP addresses and separate them using commas.

Step 3 (Optional) Set backend parameters.

Input parameters of the API are mapped to corresponding backend parameters in backend requests.

1. Click next to **Backend Parameters**, and define backend parameters. You can use one of the following methods:
   - Click **Import Input Parameter**. All the defined input parameters are automatically displayed.
   - Click **Add Backend Parameter Mapping**, and add required backend parameters.
2. Modify the mappings based on the parameters and their locations in backend requests. **Figure 3-1** highlights the backend parameters.

Figure 3-1 Configuring backend parameters



a. If you set the parameter location to **PATH**, ensure that the parameter name is the same as that defined in the backend request path.

b. The name and location of an input parameter can be different from those of the mapped backend request parameter.

NOTE

- The parameter name is not case-sensitive. It cannot start with **x-apig-** or **x-sdk-**.
- The parameter name cannot be **x-stage**.
- If you set the parameter location to **HEADER**, ensure that the parameter name does not contain underscores (_).

c. In the preceding figure, parameters **test01** and **test03** are located in the path and query positions of API requests, and their values will be received in the header of backend requests. **test02** is located in the header of API requests, and its value will be received through **test05** in the path of backend requests.

For example, **test01** is **abc**, **test02** is **def**, and **test03** is **xyz**.

API request:

```
curl -ik -H 'test02:def' -X GET https://www.example01.com/v1.0/abc?test03=xyz
```

Backend request:

```
curl -ik -H 'test01:abc' -H 'test03:xyz' -X GET https://www.example02.com/v1.0/def
```

Step 4 (Optional) Set constant parameters.

You can define constant parameters for the backend service to receive constants that are invisible to API callers. API Gateway adds constant parameters to specified positions in the request sent to the backend service.

1. Click ⌄ next to **Constant Parameters**.
2. Click **Add Constant Parameter**, and set the parameters listed in **Table 3-12**.

Table 3-12 Setting constant parameters

Parameter	Description
Name	Constant parameter name. If you set the parameter location to **PATH**, ensure that the parameter name is the same as that defined in the backend request path. **NOTE** – The parameter name is not case-sensitive. It cannot start with **x-apig-** or **x-sdk-**. – The parameter name cannot be **x-stage**. – If you set the parameter location to **HEADER**, ensure that the parameter name does not contain underscores (_).
Location	Position of the parameter in requests. The options are **PATH**, **QUERY**, and **HEADER**.
Value	Value of the parameter.

Parameter	Description
Description	Description of the constant parameter.

NOTE

- API Gateway sends requests containing constant parameters to backend services after percent-encoding of special parameter values. Ensure that the backend services support percent-encoding. For example, parameter value **[apig]** becomes **%5Bapig%5D** after percent-encoding.
- For values of path parameters, the following characters will be percent-encoded: ASCII codes 0–31, blank symbols, ASCII codes 127–255, and special characters ?></%#"[\]^`{|}
- For values of query strings, the following characters will be percent-encoded: ASCII codes 0–31, blank symbols, ASCII codes 127–255, and special characters >=<+&%#"[\]^`{|}

Step 5 (Optional) Set system parameters.

System parameters refer to runtime parameters regarding gateway running and frontend and backend authentications. The parameters are transferred to the API backend service for access control and custom authentication.

1. Click ⌄ next to **System Parameters**.
2. Click **Add System Parameter**, and set the parameters listed in **Table 3-13**.

Table 3-13 System parameters

Parameter	Description
System Parameter Type	– **Default gateway parameter**: Default parameters supported by API Gateway. – **Frontend authentication parameter**: Parameters to be displayed in the frontend custom authentication result. This option is available only if you select **Custom** for **Security Authentication** on the **Set Basic Information** page. – **Backend authentication parameter**: Parameters to be displayed in the backend custom authentication result. This option is available only if you enable **Backend Authentication** on the **Define Backend Request** page.

Parameter	Description
System Parameter Name	– Default gateway parameters include **sourceIp** (source IP address used to call the API), **stage** (environment in which the API is called), **apiId** (ID of the API), **appId** (ID of the app used to call the API), and **requestId** (ID of the request initiated to call the API). – Ensure that the frontend and backend authentication parameters are consistent with the return result parameters defined for the corresponding custom authorizer function. For details about how to create a custom authorizer function and obtain returned result parameters, see *API Gateway Developer Guide*.
Backend Parameter Name	Name of the backend parameter to which the system parameter will be mapped. **NOTE** – The parameter name is not case-sensitive. It cannot start with **x-apig-** or **x-sdk-**. – The parameter name cannot be **x-stage**. – If you set the parameter location to **HEADER**, ensure that the parameter name does not contain underscores (_).
Backend Parameter Location	Position of the backend parameter in requests.
Description	Description of the system parameter.

Step 6 Click **Next**.

----End

Defining Responses

Step 1 On the **Define Response** page, set the parameters listed in **Table 3-14**.

Table 3-14 Defining responses

Parameter	Description
Example Success Response	An example of a response returned when the API is called successfully.
Example Failure Response	An example of a response returned when the API fails to be called.

Step 2 Click **Finish**.

After the API is created, click its name in the API list to view details.

----End

Follow-Up Operations

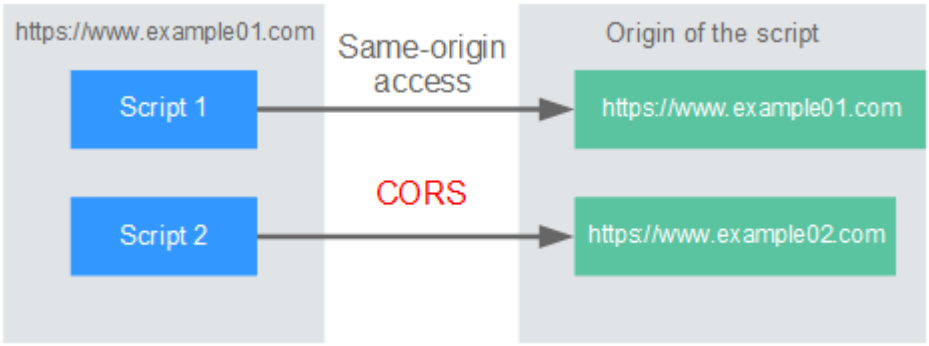
After creating an API, verify it by following the procedure in **Debugging an API**.

3.2.2 CORS

What Is CORS?

For security reasons, browsers restrict cross-origin requests initiated from within scripts. This means that a web application can only request resources from its origin. The CORS mechanism allows browsers to send XMLHttpRequest to servers in other domains and request access to the resources there.

Figure 3-2 Process flow of the CORS mechanism



There are two types of CORS requests:

- **Simple requests**

  Simple requests must meet the following conditions:

  a. The request method is HEAD, GET, or POST.

  b. The request header contains only the following fields:

     - Accept
     - Accept-Language
     - Content-Language
     - Last-Event-ID
     - Content-Type (**application/x-www-form-urlencoded**, **multipart/form-data**, or **text/plain**)

  In the header of a simple request, browsers automatically add the **Origin** field to specify the origin (including the protocol, domain, and port) of the request. After receiving such a request, the target server determines whether the request is safe and can be accepted based on the origin. If the server sends a response containing the **Access-Control-Allow-Origin** field, the server accepts the request.

- **Not-so-simple requests**

  Requests that do not meet the conditions for simple requests are not-so-simple requests.

Before sending a not-so-simple request, browsers send an HTTP preflight request to the target server to confirm whether the origin the web page is loaded from is in the allowed origin list, and to confirm which HTTP request methods and header fields can be used. If the preflight request is successful, browsers send simple requests to the server.

Configuring CORS

CORS is disabled by default. To enable CORS for an API, perform the operations described in this section.

- **Simple CORS requests**

  When creating an API, enable CORS on the API request configuration page. For more information, see **Simple Request**.

- **Not-so-simple CORS requests**

  NOTICE

  If your API will receive not-so-simple requests, **create another API that will be accessed using the OPTIONS method** in the same group as the target API to receive preflight requests.

  Follow this procedure to define the preflight request API. For more information, see **Not-So-Simple Request**.

  a. On the **Set Basic Information** page, select **None** to skip over security authentication.
  b. On the **Define API Request** page, perform the following settings:
     - **Protocol**: The same protocol used by the API with CORS enabled.
     - **Path**: Enter a slash (/).
     - **Method**: Select **OPTIONS**.
     - **CORS**: Enabled.
  c. Select the **Mock** backend type.

Simple Request

When creating an API that will receive simple requests, **enable CORS** for the API.

Scenario 1: If CORS is enabled and the response from the backend does not contain a CORS header, API Gateway handles requests from any domain, and returns the **Access-Control-Allow-Origin** header. For example:

Request sent by a browser and containing the Origin header field:

```
GET /simple HTTP/1.1
Host: www.test.com
Orgin: http://www.cors.com
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Accept: application/json
Date: Tue, 15 Jan 2019 01:25:52 GMT
```

Origin: This field is required to specify the origin (**http://www.cors.com** in this example) of the request. API Gateway and the backend service determine based on the origin whether the request is safe and can be accepted.

Response sent by the backend service:

```
HTTP/1.1 200 OK
Date: Tue, 15 Jan 2019 01:25:52 GMT
Content-Type: application/json
Content-Length: 16
Server: api-gateway

{"status":"200"}
```

Response sent by API Gateway:

```
HTTP/1.1 200 OK
Date: Tue, 15 Jan 2019 01:25:52 GMT
Content-Type: application/json
Content-Length: 16
Server: api-gateway
X-Request-Id: 454d689fa69847610b3ca486458fb08b
Access-Control-Allow-Origin: *

{"status":"200"}
```

Access-Control-Allow-Origin: This field is required. The asterisk (*) means that API Gateway handles requests sent from any domain.

Scenario 2: If CORS is enabled and the response from the backend contains a CORS header, the header will overwrite that added by API Gateway. The following messages are used as examples:

Request sent by a browser and containing the Origin header field:

```
GET /simple HTTP/1.1
Host: www.test.com
Orgin: http://www.cors.com
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Accept: application/json
Date: Tue, 15 Jan 2019 01:25:52 GMT
```

Origin: This field is required to specify the origin (**http://www.cors.com** in this example) of the request. API Gateway and the backend service determine based on the origin whether the request is safe and can be accepted.

Response sent by the backend service:

```
HTTP/1.1 200 OK
Date: Tue, 15 Jan 2019 01:25:52 GMT
Content-Type: application/json
Content-Length: 16
Server: api-gateway
Access-Control-Allow-Origin: http://www.cors.com

{"status":"200"}
```

Access-Control-Allow-Origin: Indicates that the backend service accepts requests sent from **http://www.cors.com**.

Response sent by API Gateway:

```
HTTP/1.1 200 OK
Date: Tue, 15 Jan 2019 01:25:52 GMT
Content-Type: application/json
Content-Length: 16
Server: api-gateway
```

```
X-Request-Id: 454d689fa69847610b3ca486458fb08b
Access-Control-Allow-Origin: http://www.cors.com

{"status":"200"}
```

The CORS header in the backend response overwrites that in API Gateway's response.

Not-So-Simple Request

When creating an API that will receive not-so-simple requests, enable CORS for the API by following the instructions in **Configuring CORS**, and create another API that will be accessed using the OPTIONS method.

The request parameters of an API accessed using the OPTIONS method must be set as follows:

- **API Group**: The same group to which the API with CORS enabled belongs.
- **Security Authentication**: Select **None**. No authentication is required for requests received by the new API no matter which security authentication mode has been selected.
- **Protocol**: The same protocol used by the API with CORS enabled.
- **Path**: Enter a slash (/) or select the path that has been set for or matches the API with CORS enabled.
- **Method**: Select **OPTIONS**.
- **CORS**: Enabled.

The following are example requests and responses sent to or from a mock backend.

Request sent from a browser to an API that is accessed using the OPTIONS method:

```
OPTIONS /HTTP/1.1
User-Agent: curl/7.29.0
Host: localhost
Accept: */*
Origin: http://www.cors.com
Access-Control-Request-Method: PUT
Access-Control-Request-Headers: X-Sdk-Date
```

- **Origin**: This field is required to specify the origin from which the request has been sent.
- **Access-Control-Request-Method**: This field is required to specify the HTTP methods to be used by the subsequent simple requests.
- **Access-Control-Request-Headers**: This field is optional and used to specify the additional header fields in the subsequent simple requests.

Response sent by the backend: none

Response sent by API Gateway:

```
HTTP/1.1 200 OK
Date: Tue, 15 Jan 2019 02:38:48 GMT
Content-Type: application/json
Content-Length: 1036
Server: api-gateway
X-Request-Id: c9b8926888c356d6a9581c5c10bb4d11
Access-Control-Allow-Origin: *
```

```
Access-Control-Allow-Headers: X-Stage,X-Sdk-Date,X-Sdk-Nonce,X-Proxy-Signed-Headers,X-Sdk-Content-Sha256,X-Forwarded-For,Authorization,Content-Type,Accept,Accept-Ranges,Cache-Control,Range
Access-Control-Expose-Headers: X-Request-Id,X-Apig-Latency,X-Apig-Upstream-Latency,X-Apig-RateLimit-Api,X-Apig-RateLimit-User,X-Apig-RateLimit-App,X-Apig-RateLimit-Ip,X-Apig-RateLimit-Api-Allenv
Access-Control-Allow-Methods: GET,POST,PUT,DELETE,HEAD,OPTIONS,PATCH
Access-Control-Max-Age: 172800
```

- **Access-Control-Allow-Origin**: This field is required. The asterisk (*) means that API Gateway handles requests sent from any domain.
- **Access-Control-Allow-Headers**: This field is required if it is contained in the request. It indicates all header fields that can be used during cross-origin access.
- **Access-Control-Expose-Headers**: This is the response header fields that can be viewed during cross-region access.
- **Access-Control-Allow-Methods**: This field is required to specify which HTTP request methods the API Gateway supports.
- **Access-Control-Max-Age**: This field is optional and used to specify the length of time (in seconds) during which the preflight result remains valid. No more preflight requests will be sent within the specified period.

Request sent by a browser and containing the Origin header field:

```
PUT /simple HTTP/1.1
Host: www.test.com
Orgin: http://www.cors.com
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Accept: application/json
Date: Tue, 15 Jan 2019 01:25:52 GMT
```

Response sent by the backend:

```
HTTP/1.1 200 OK
Date: Tue, 15 Jan 2019 01:25:52 GMT
Content-Type: application/json
Content-Length: 16
Server: api-gateway

{"status":"200"}
```

Response sent by API Gateway:

```
HTTP/1.1 200 OK
Date: Tue, 15 Jan 2019 01:25:52 GMT
Content-Type: application/json
Content-Length: 16
Server: api-gateway
X-Request-Id: 454d689fa69847610b3ca486458fb08b
Access-Control-Allow-Origin: *

{"status":"200"}
```

3.2.3 Debugging an API

Scenario

After creating an API, debug it on the API Gateway console by setting HTTP headers and body parameters to verify whether the API is running normally.

NOTE

- APIs with backend request paths containing variables cannot be debugged.
- If an API has been bound with a request throttling policy, the policy will not work during debugging of the API.

Prerequisites

- You have created an API group and API.
- You have set up the backend service of the API.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **APIs**.

Step 4 Debug an API. You can use one of the following methods:

- In the **Operation** column of the API you want to debug, choose **More** > **Debug**.
- Click the name of the target API, and click **Debug** in the upper right corner of the displayed API details page.

On the left side, set the API request parameters listed in **Table 3-15**. On the right side, view the API request and response information after you click **Send Request**.

Table 3-15 Parameters for debugging an API

Parameter	Description
Protocol	This parameter can be modified only if you set **Protocol** to **HTTP&HTTPS** for the API.
Method	This parameter can be modified only if you set **Method** to **ANY** for the API.
Suffix	You can define a path only if you have set **Matching** to **Prefix match** for the API.
Path	Request path of the API.
Path Parameters	This parameter can be modified only if you have defined path parameters (such as **{test}**) for the API.
Headers	HTTP headers and values.
Query Strings	Query string parameters and values.
Body	This parameter can be modified only if you set **Method** to **PATCH**, **POST**, or **PUT** for the API.

NOTE

The fields displayed on the debugging page vary according to the request type.

Step 5 After setting request parameters, click **Send Request**.

The box on the lower right displays the response of the API request.

- If the debugging is successful, the HTTP status code **200** and response details are displayed.
- If the request fails to be sent, an HTTP status code **4xx** or **5xx** is displayed. For details, see **Error Codes**.

Step 6 You can send more requests with different parameters and values to verify the API.

NOTE

To modify the API configurations, click **Edit** in the upper right corner, and modify the parameters on the **Edit API** page.

----End

Follow-Up Operations

After the API is successfully debugged, **publish** the API in a specific environment so that the API can be called by users. To ensure security of the API, create request throttling policies (see **Creating a Request Throttling Policy**), access control policies (**Creating an Access Control Policy**), and signature keys (**Creating and Using a Signature Key**) for the API.

3.2.4 Authorizing Apps to Call an API

Scenario

APIs using app authentication can only be called by apps that have been authorized to call them.

NOTE

- APIs can be called only after they have been published.
- You can authorize apps only to call APIs that use app authentication.

Prerequisites

- You have created an API group and API.
- (Optional) You have created an environment.
- You have created an app.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **APIs**.

Step 4 Authorize apps to call an API. You can use one of the following methods:

- In the **Operation** column of the target API, click **Authorize App**, and then click **Select App**.
- Select the target API, click **Authorize App** over the API list, and then click **Select App**.
- Authorize apps through the API details page.
  a. Click the name of the target API.
  b. Click the **Authorization** tab.
  c. Click **Select App**.

NOTE

To authorize an app to access multiple APIs, select the APIs, and click **Authorize App**. Click **Select App**, select the app you wish to authorize, and click **OK**. You can grant access to a maximum of 1000 APIs at a time.

Step 5 Select an environment, search for and select desired apps, and click **OK**.

Step 6 After the authorization is complete, view the authorized apps on the **Authorization** tab page or the **Authorize App** page.

NOTE

If an app does not need to call the API, click **Cancel Authorization** in the row containing the app to unbind it.

----End

Follow-Up Operations

After you authorize an app to call an API, the API can be called using SDKs of different programming languages.

3.2.5 Publishing an API

Scenario

APIs can be called only after they have been published in an environment. You can publish APIs in different environments. API Gateway allows you to view the publication history (such as the version, description, time, and environment) of each API, and supports rollback of APIs to different historical versions.

NOTE

- If you modify a published API, you must publish it again for the modifications to take effect in the environment in which the API has been published.
- A maximum of 10 publication records of an API are retained in an environment.

Prerequisites

- You have created an API group and API.

- You have created an environment.

Publishing an API

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **APIs**.

Step 4 Publish an API. You can use one of the following methods:

- Click **Publish** in the row containing the API you want to publish.
- Click the name of the target API, and click **Publish** in the upper right corner of the displayed API details page.

NOTE

To publish multiple APIs, select the APIs, and click **Publish**. You can publish a maximum of 1000 APIs at a time.

Step 5 Select the environment where the API will be published, and enter a description.

NOTE

- If the API has already been published in the environment, publishing it again will overwrite its definition in that environment.
- If there is no environment that meets your requirements, create a new one.

Step 6 Click **Publish**.

----End

Viewing Publication History

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **APIs**.

Step 4 Click the name of the target API.

Step 5 Click the **Publication History** tab.

The publication history of the API is displayed.

Step 6 Click **View Details** in the **Operation** column of a version.

The **View Details** dialog box displays the basic information, frontend and backend request information, input and constant parameters, parameter mappings, and example responses of the API.

Step 7 To roll back the API to a historical version, click **Switch Version** in the row containing the target version, and click **Yes**.

If "current version" is displayed next to the target version, the rollback was successful.

When the API is called, configuration of the current version is used instead of the previously saved configuration.

For example, an API was published in the RELEASE environment on August 1, 2018. On August 20, 2018, the API was published in the same environment after modification. If the version published on August 1 is set as the current version, configuration of this version will be used when the API is called.

----End

3.2.6 Taking an API Offline

Scenario

You can remove APIs that you do not need from the environments where the APIs have been published.

NOTICE

This operation will cause the APIs to be inaccessible in the environments. Ensure that you have notified users before this operation.

Prerequisites

- You have created an API group and API.
- You have published the API.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **APIs**.

Step 4 Take the API offline. You can use one of the following methods:

- In the **Operation** column of the target API, choose **More** > **Take Offline**.
- Click the name of the target API, and click **Take Offline** in the upper right corner of the API details page.

NOTE

To take multiple APIs offline, select the APIs, and click **Take Offline**. You can take a maximum of 1000 APIs offline at a time.

Step 5 Select the environment from which you want to take the API offline, and click **Yes**.

----End

Follow-Up Operations

After taking an API offline, delete it based on the instructions provided in **Deleting an API**.

3.2.7 Deleting an API

Scenario

You can delete published APIs you no longer require.

NOTICE

- Deleted APIs cannot be accessed by apps or users who were using the APIs, so make sure you notify users before the deletion.
- Published APIs must be first taken offline and then deleted.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **APIs**.

Step 4 Delete the API. You can use one of the following methods:

- In the **Operation** column of the API you want to delete, choose **More** > **Delete**.
- Click the name of the target API, and click **Delete** in the upper right corner of the displayed API details page.

NOTE

To delete multiple APIs, select the APIs, and click **Delete**. You can delete a maximum of 1000 APIs at a time.

Step 5 Enter **DELETE** and click **Yes**.

----End

3.2.8 Importing APIs

Scenario

API Gateway supports the import of APIs defined in accordance with Swagger 2.0 into an existing or a new API group. Swagger is a suite of open-source API developer tools that comply with OpenAPI Specification and help you design, build, document, and use REST APIs.

You can import APIs individually or in batches depending on the number of APIs contained in a Swagger file.

Prerequisites

- The API Swagger file to be imported is available and already has extended API definitions supplemented. For more information, see .

- You have sufficient quotas of API groups and APIs.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **APIs**.

Step 4 Click **Import API**.

Step 5 Set the parameters listed in **Table 3-16**.

Table 3-16 Parameters for importing APIs

Parameter	Description
Import	Options: • **New group**: Import APIs to a new API group. If you select this option, the system automatically creates a new API group and imports the APIs into this group. • **Existing group**: Import APIs to an existing API group. If you select this option, the system adds the APIs to the selected API group while retaining the existing APIs in the API group.
API group	Select an API group if you set **Import** to **Existing group**.
Basic Definition Overwrite	Determine whether to overwrite the basic definition of an existing API if the name of the API is the same as that of an imported API. This parameter is available only if you set **Import** to **Existing group**.
Extended Definition Overwrite	Determine whether to overwrite the extended definition (such as access control and request throttling policies) of an existing API if the extended definition name of the API is the same as that of an imported API.

Step 6 In the **Parameter Import** area, click **File** and select a file to import.

YAML and JSON files are supported. You can preview the API content to be imported on the **Import API** page.

Step 7 (Optional) Configure global settings for the APIs to be imported.

You can configure the global settings for the APIs, such as frontend and backend requests, or modify other parameters of the APIs.

Step 8 Click **Import Now** to import the APIs.

NOTE

Imported APIs must be manually published so that they become available for users to access.

----End

Follow-Up Operations

Publish the imported API in an environment so that it can be called by users.

3.2.9 Exporting APIs

Scenario

You can export APIs one by one or in batches as JSON or YAML files.

Prerequisites

You have created an API group and API.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 Click **Export API**.

Step 4 Set the parameters listed in **Table 3-17**.

Table 3-17 Parameters for exporting APIs

Parameter	Description
API Group	Select the API group from which APIs will be exported.
Environment	Select the environment where the APIs to be exported have been published.
APIs	By default, all APIs in the API group that have been published in the selected environment are exported. To export only specific APIs, click **Select API**, and specify the APIs you want to export.

Parameter	Description
API Definition	• **Basic**: The basic definition of an API is composed of the request and response definitions. It does not include the backend definition. The request definition includes both standard and extended Swagger fields. • **Full**: The full definition of an API is composed of the request, backend, and response definitions. • **Extended**: The extended definition of an API is composed of the request, backend, and response definitions as well as the request throttling policy, access control policy, and other configurations of the API.
Format	Export APIs in **JSON** or **YAML** format.
Version	Set the version of the APIs to be exported. If you do not specify a version, the version will be set as the current date and time.

Step 5 Click **Export**.

The export result is displayed on the right.

----End

3.3 Request Throttling

3.3.1 Creating a Request Throttling Policy

Scenario

Request throttling controls the number of times an API can be called within a time period to protect backend services.

To provide stable, uninterrupted services, you can create request throttling policies to control the number of calls made to your APIs.

Request throttling policies take effect for an API only if they have been bound to the API.

NOTE

- You can create a maximum of 30 request throttling policies.
- An API can be bound with only one request throttling policy for a given environment, but each request throttling policy can be bound to multiple APIs.
- For APIs with no request throttling policies bound, the default throttling limit (200 calls per second) is used.

Prerequisites

You have **published the API** to which you want to bind a request throttling policy.

Creating a Request Throttling Policy

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **Request Throttling**.

Step 4 Click **Create Request Throttling Policy**, and set the parameters listed in **Table 3-18**.

Table 3-18 Parameters for creating a request throttling policy

Parameter	Description
Name	Request throttling policy name.
Type	API-based or API-shared request throttling. • **API-based**: Request throttling is based on every API to which the policy is bound. • **API-shared**: Request throttling is based on all APIs as a whole to which the policy is bound.
Period	For how long you want to limit the number of API calls. This parameter can be used together with the following parameters: • **Max. API Requests**: Limit the maximum number of times an API can be called within a specific period. • **Max. User Requests**: Limit the maximum number of times an API can be called by a user within a specific period. • **Max. App Requests**: Limit the maximum number of times an API can be called by an app within a specific period. • **Max. IP Address Requests**: Limit the maximum number of times an API can be called by an IP address within a specific period.
Max. API Requests	The maximum number of times each bound API can be called within the specified period. This parameter must be used together with **Period**.
Max. User Requests	The maximum number of times each bound API can be called by a user within the specified period. **This limit only applies to APIs that are accessed through IAM authentication.** • The value of this parameter cannot exceed that of **Max. API Requests**. • This parameter must be used together with **Period**. • If there are many users under your account that access an API, the request throttling limits of the API will apply to all these users.

Parameter	Description
Max. App Requests	The maximum number of times each bound API can be called by an app within the specified period. **This limit only applies to APIs that are accessed through app authentication.** • The value of this parameter cannot exceed that of **Max. User Requests**. • This parameter must be used together with **Period**.
Max. IP Address Requests	The maximum number of times each bound API can be called by an IP address within the specified period. • The value of this parameter cannot exceed that of **Max. API Requests**. • This parameter must be used together with **Period**.
Description	Description of the request throttling policy.

Step 5 Click **OK**.

After the policy is created, it is displayed on the **Request Throttling** page. You can bind this policy to APIs to throttle API requests.

----End

Binding a Request Throttling Policy to an API

Step 1 Go to the page for binding a request throttling policy to an API. You can use one of the following methods:

- In the **Operation** column of the request throttling policy to be bound, click **Bind to API**, and then click **Select API**.
- Click the name of the target request throttling policy, and click **Select API** on the **APIs** tab page.

Step 2 Specify an API group, environment, and API name keyword to search for the desired API.

Step 3 Select the API and click **OK**.

NOTE

If a request throttling policy is no longer needed for an API, you can unbind it. To unbind a request throttling policy from multiple APIs, select the APIs, and click **Unbind**. You can unbind a request throttling policy from a maximum of 1000 APIs at a time.

----End

Follow-Up Operations

To control the maximum number of API calls received from a specific app or tenant, specify the app or tenant to exclude by referring to **Adding an Excluded App or Tenant**. If an app is excluded in a request throttling policy, any threshold configured for that app takes precedence over the request throttling policy. The API and user request limits of this policy are still valid. If a tenant is excluded in a

request throttling policy, any threshold configured for that tenant will be applied. The API and app request limits of this policy are still valid.

3.3.2 Deleting a Request Throttling Policy

Scenario

You can delete request throttling policies you no longer require.

Prerequisites

You have created a request throttling policy.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **Request Throttling**.

Step 4 Delete a request throttling policy. You can use one of the following methods:

- In the **Operation** column of the request throttling policy you want to delete, click **Delete**.
- Click the name of the target request throttling policy, and click **Delete** in the upper right corner of the displayed request throttling policy details page.

NOTE

- If a request throttling policy has been bound to an API, unbind the policy and then delete it. To unbind a request throttling policy, go to the policy details page, click **Unbind** in the row that contains the API from which you want to unbind the policy, and click **Yes**.
- To delete multiple request throttling policies, select the policies, and click **Delete**. You can delete a maximum of 1000 request throttling policies at a time.

Step 5 Click **Yes**.

----End

3.3.3 Adding an Excluded App or Tenant

Scenario

If you want to control the number of API calls received from a specific app or tenant, add an excluded app or tenant to a request throttling policy.

Prerequisites

You have created an app or obtained an app ID of another account or an account ID.

Adding an Excluded App

Step 1 Log in to the management console.

Step 2 Click in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **Request Throttling**.

Step 4 Click the name of the target request throttling policy.

Step 5 On the displayed request throttling policy details page, click the **Excluded Apps** tab.

Step 6 Click **Select Excluded App**.

Step 7 Select an app to exclude. You can use one of the following methods:

- To select an existing app, click **Existing**, select an app, and enter a threshold.
- To select an app of other tenants, click **Cross-tenant**, and enter the app ID and a threshold.

NOTE

The threshold must be a positive integer and cannot exceed the value of **Max. API Requests**.

----End

Adding an Excluded Tenant

Step 1 Log in to the management console.

Step 2 Hover the mouse pointer over the username and choose **My Credentials** from the drop-down list.

Step 3 On the **My Credentials** page, view the account ID and project ID.

Step 4 Click in the upper left corner and choose **API Gateway**.

Step 5 In the navigation pane, choose **API Publishing** > **Request Throttling**.

Step 6 Click the name of the target request throttling policy.

Step 7 Click the **Excluded Tenants** tab.

Step 8 Click **Select Excluded Tenant**.

Step 9 In the **Select Excluded Tenant** dialog box, set the parameters listed in **Table 3-19**.

Table 3-19 Excluded tenant configuration

Parameter	Description
Account ID	Account ID or project ID obtained in **Step 3**. • Enter a project ID if you will bind or have bound this policy to an API that uses app authentication. • Enter an account ID if you will bind or have bound this policy to an API that uses IAM authentication.
Threshold	The maximum number of times an API can be called by the tenant within a specified period. The value of this parameter cannot exceed that of **Max. API Requests**.

Step 10 Click **OK**.

NOTE

Excluded tenant thresholds take precedence over the value of **Max. User Requests**.

For example, suppose a request throttling policy is configured, with **Max. API Requests** being **10**, **Max. User Requests** being **3**, **Period** being 1 minute, and two excluded tenants (max. **2** API requests for tenant A and max. **4** API requests for tenant B). If the request throttling policy is bound to an API, tenants A and B can access the API 2 and 4 times within 1 minute, respectively.

----End

3.3.4 Removing an Excluded App or Tenant

Scenario

You can remove excluded apps or tenants from a request throttling policy. This section takes an excluded app as an example.

Prerequisites

- You have created a request throttling policy.
- You have already added an excluded app or tenant to the request throttling policy.

Removing an Excluded App

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **Request Throttling**.

Step 4 Click the name of the target request throttling policy.

Step 5 Click the **Excluded Apps** tab on the displayed request throttling policy details page.

Step 6 In the **Operation** column of the app you want to remove, click **Remove**.

Step 7 Click **Yes**.

----End

Removing an Excluded Tenant

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **Request Throttling**.

Step 4 Click the name of the target request throttling policy.

Step 5 Click the **Excluded Tenants** tab.

Step 6 In the **Operation** column of the tenant you want to remove, click **Remove**.

Step 7 Click **Yes**.

----End

3.4 Access Control

3.4.1 Creating an Access Control Policy

Scenario

Access control policies are a type of security measures provided by API Gateway. You can use them to allow or deny API access from specific IP addresses or accounts.

Access control policies take effect for an API only if they have been bound to the API.

NOTE

- You can create a maximum of 100 access control policies.
- Each API can be bound with only one access control policy for a given environment, but each access control policy can be bound to multiple APIs.

Creating an Access Control Policy

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **Access Control**.

Step 4 Click **Create Access Control Policy**.

Step 5 In the **Create Access Control Policy** dialog box, set the parameters listed in **Table 3-20**.

Table 3-20 Parameters for creating an access control policy

Parameter	Description
Name	Access control policy name.
Restriction Type	Type of the source from which API calls are to be controlled. • **IP address**: Specify IP addresses and IP address ranges that are allowed or not allowed to access an API. • **Account name**: Specify names of the accounts that are allowed or not allowed to access an API.
Effect	Options: **Allow** and **Deny**. Use this parameter along with **Restriction Type** to control the access of certain IP addresses or accounts to an API.
IP Address	IP addresses and IP address ranges that are allowed or not allowed to access an API You need to set this parameter only if you have set **Restriction Type** to **IP address**. **NOTE** You can set a maximum of 100 IP addresses respectively to allow or deny access.
Account Names	Names of the accounts that are allowed or not allowed to access an API. **This parameter only applies to APIs that are accessed through IAM authentication.** You need to set this parameter only if you have set **Restriction Type** to **Account name**. You can enter multiple account names and separate them with commas, for example, **aaa,bbb**. **NOTE** API Gateway performs access control on accounts, not IAM users created using accounts.

Step 6 Click **OK**. You can bind the policy to APIs to control API access.

----End

Binding an Access Control Policy to an API

Step 1 Go to the page for binding an access control policy to an API. You can use one of the following methods:

- In the **Operation** column of the access control policy to be bound, click **Bind to API**, and then click **Select API**.
- Click the name of the target access control policy, and click **Select API**.

Step 2 Specify an API group, environment, and API name keyword to search for the desired API.

Step 3 Select the API and click **OK**.

NOTE

If an access control policy is no longer needed for an API, you can unbind it from that API. To unbind an access control policy from multiple APIs, select the APIs, and click **Unbind**. You can unbind a request throttling policy from a maximum of 1000 APIs at a time.

----End

3.4.2 Deleting an Access Control Policy

Scenario

You can delete access control policies you no longer require.

Prerequisites

You have created an access control policy.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **Access Control**.

Step 4 Delete an access control policy using one of the following methods:

- In the **Operation** column of the access control policy you want to delete, click **Delete**.
- Click the name of the target access control policy, and click **Delete** in the upper right corner of the displayed access control policy details page.

NOTE

- If an access control policy has been bound to APIs, unbind it and then delete it.
- To delete multiple access control policies, select the policies, and click **Delete**. You can delete a maximum of 1000 access control policies at a time.

Step 5 Click **Yes**.

----End

3.5 Environment Management

3.5.1 Creating an Environment and Environment Variable

Scenario

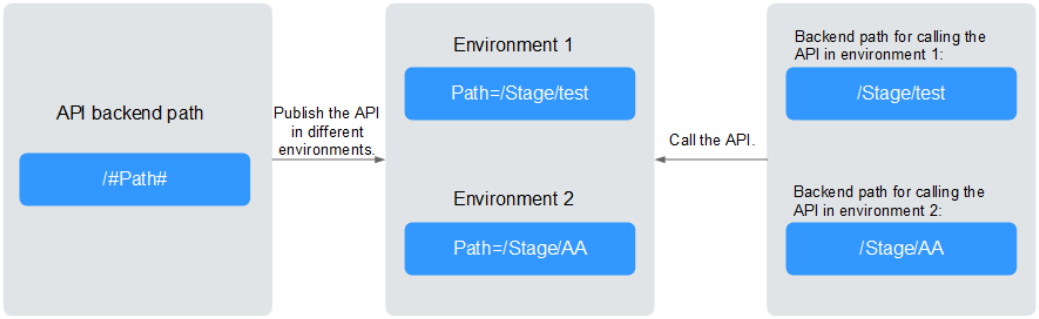
An API can be called in different environments, such as production, testing, and development environments. RELEASE is the default environment provided by API

Gateway. You can define environment variables to allow an API to be called in different environments.

Environment variables are manageable and specific to environments. You can create variables in different environments to call different backend services using the same API.

For variables you define during API creation, you must create corresponding variables and values. For example, variable **Path** is defined for an API, and two variables with the same name are created and assigned values **/Stage/test** and **/Stage/AA** in environments 1 and 2, respectively. If the API is published and called in environment 1, the path **/Stage/test** is used. If the API is published and called in environment 2, the path **/Stage/AA** is used.

Figure 3-3 Use of environment variables



NOTE

You can create a maximum of 10 environments and a maximum of 50 variables for an API group in an environment.

Prerequisites

You have **created an API group**.

Creating an Environment

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **Environments**.

Step 4 Click **Create Environment**, and set the parameters listed in **Table 3-21**.

Table 3-21 Environment information

Parameter	Description
Name	Environment name.
Description	Description of the environment.

Step 5 Click **OK**.

After the environment is created, it is displayed in the environment list.

----End

Accessing an Environment

You can call an API in the RELEASE environment by using a RESTful API. To access the API in other environments, add the **X-Stage** header to the request to specify an environment name. For example, add **X-Stage:DEVELOP** to the request header to access an API in the **DEVELOP** environment.

NOTE

API Gateway does not support API debugging using environment variables.

Creating an Environment Variable

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **API Groups**.

Step 4 Create a variable. You can use one of the following methods:

- Click the name of the target API group, and click the **Variables** tab on the displayed API group details page.
- In the **Operation** column of the target API group, choose **More** > **Manage Variable**.

Step 5 Select an environment from the **Environment** drop-down list, and click **Create Variable**.

Step 6 Set the parameters listed in **Table 3-22**.

Table 3-22 Parameters for creating an environment variable

Parameter	Description
Name	Name of the variable you want to create. Ensure that the name is the same as the name of the variable defined for the API.
Value	The path to be used in the selected environment.

Step 7 Click **OK**.

NOTE

If a variable is not needed, click **Delete** in the row containing the variable to delete it.

Environment variable names and values will be displayed in plain text in API requests. Do not include sensitive information in the variable names and values.

----End

Follow-Up Operations

After creating an environment and variable, **publish APIs** in the environment so that they can be called by API callers.

3.5.2 Deleting an Environment

Scenario

You can delete environments you no longer require.

Prerequisites

You have created an environment.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **Environments**.

Step 4 In the **Operation** column of the environment you want to delete, click **Delete**.

NOTE

You can delete an environment only if no APIs have been published in the environment.

Step 5 Click **Yes**.

----End

3.6 Signature Key Management

3.6.1 Creating and Using a Signature Key

Scenario

Signature keys are used by backend services to verify the identity of API Gateway.

A signature key consists of a key and secret, and can be used only after being bound to an API. When an API bound with a signature key is called, API Gateway adds signature details to the API request. The backend service of the API signs the request in the same way, and verifies the identity of API Gateway by checking whether the signature is consistent with that in the **Authorization** header sent by API Gateway.

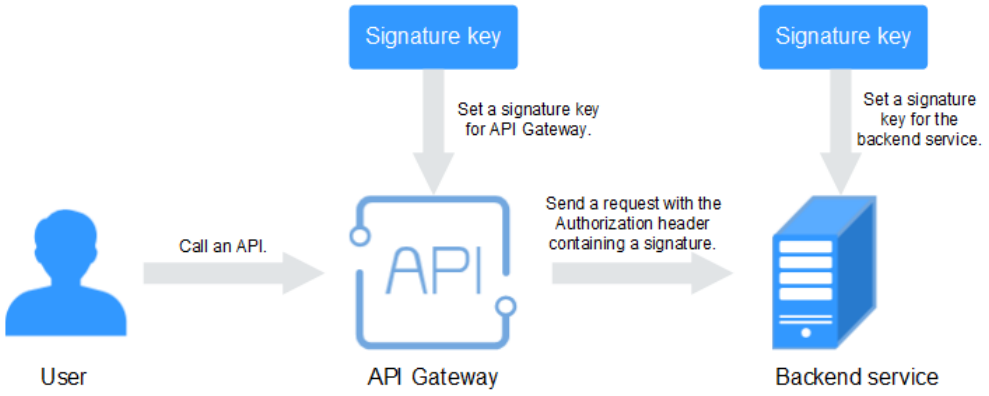
NOTE

- You can create a maximum of 30 signature keys.
- An API can only be bound with one signature key in a given environment, but each signature key can be bound to multiple APIs.

Procedure

1. Create a signature key on the API Gateway console.
2. Bind the signature key to an API.
3. API Gateway sends signed requests containing a signature in the **Authorization** header to the backend service. The backend service can use different programming languages (such as Java, Go, Python, JavaScript, C#, PHP, C++, C, and Android) to sign each request, and check whether the two signatures are consistent.

Figure 3-4 Signature key process flow



Creating a Signature Key

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **Signature Keys**.

Step 4 Click **Create Signature Key**.

Step 5 In the **Create Signature Key** dialog box, set the parameters listed in **Table 3-23**.

Table 3-23 Parameters for creating a signature key

Parameter	Description
Name	Signature key name.
Key	Combined with **Secret** to form a signature key pair.
Secret	Combined with **Key** to form a signature key pair.
Confirm Secret	Enter the secret again.

Step 6 Click **OK**.

----End

Binding a Signature Key to an API

Step 1 In the navigation pane, choose **API Publishing** > **Signature Keys**.

Step 2 Bind a signature key to an API. You can use one of the following methods:

- In the **Operation** column of the signature key to be bound to an API, click **Bind to API**.
- Click the name of the target signature key.

Step 3 Click **Select API**.

Step 4 Specify an API group, environment, and API name keyword to search for the desired API.

Step 5 Select the API and click **OK**.

NOTE

If a signature key is no longer needed for an API, unbind it from the API.

----End

Verifying the Signing Result

Sign each backend request by following the instructions in chapter "Creating Signatures for Backend Requests" of the *Developer Guide*, and check whether the backend signature is consistent with the signature in the **Authorization** header of the API request.

3.6.2 Deleting a Signature Key

Scenario

You can delete signature keys you no longer require.

Prerequisites

You have created a signature key.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **Signature Keys**.

Step 4 Delete a signature key. You can use one of the following methods:

- In the **Operation** column of the signature key you want to delete, click **Delete**.
- Click the name of the target signature key, and click **Delete** in the upper right corner of the displayed signature key details page.

NOTE

If the signature key has been bound to any APIs, unbind it and then delete it.

Step 5 Click **Yes**.

----End

3.7 VPC Channel Management

3.7.1 Creating a VPC Channel

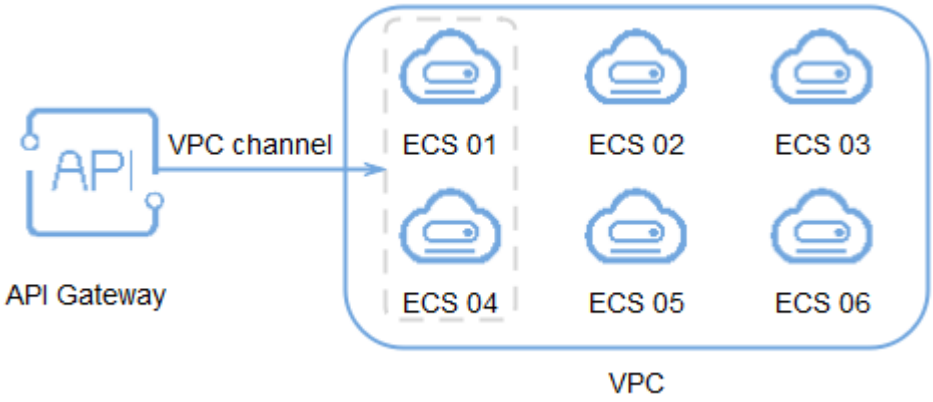
Scenario

VPC channels allow services deployed in VPCs to be accessed through their subnets, lowering latency and balancing loads of backend services.

After creating a VPC channel, you can configure it for an API with an HTTP/HTTPS backend service. You can create a maximum of 30 VPC channels.

For example, six ECSs have been deployed in a VPC, and a VPC channel has been created to reach ECS 01 and ECS 04. API Gateway can access these two ECSs through the VPC channel.

Figure 3-5 Accessing ECSs in a VPC channel through API Gateway



Prerequisites

You have created cloud servers.

Creating a Fast Channel

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **VPC Channels**.

Step 4 Click **Create VPC Channel**, and set the parameters listed in **Table 3-24**.

Table 3-24 Parameters for creating a VPC channel

Parameter	Description
Name	VPC channel name.
Port	The host port of the VPC channel, that is, the port of the backend service. Range: 1–65535.
Routing Algorithm	The algorithm to be used to forward requests to cloud servers you select. The following routing algorithms are available: • **WRR**: weighted round robin • **WLC**: weighted least connection • **SH**: source hashing • **URI hashing**
Protocol	The protocol used to perform health checks on cloud servers associated with the VPC channel. Options: • TCP • HTTP • HTTPS Default value: **TCP**.
Path	The destination path for health checks. Set this parameter only when **Protocol** is not set to **TCP**.
Check Port	The destination port for health checks. By default, the port of the VPC channel will be used.
Healthy Threshold	The number of consecutive successful checks required for a cloud server to be considered healthy. Range: 2–10. Default value: **2**.
Unhealthy Threshold	The number of consecutive failed checks required for a cloud server to be considered unhealthy. Range: 2–10. Default value: **5**.
Timeout (s)	The timeout used to determine whether a health check has failed. Unit: s. Range: 2–30. Default value: **5**.
Interval (s)	The interval between consecutive checks. Unit: s. Range: 5–300. Default value: **10**.
Response Codes	The HTTP codes used to check for a successful response from a target. Set this parameter only when **Protocol** is not set to **TCP**.

Step 5 Click **Next**.

Step 6 Click **Select Cloud Server**.

Step 7 Select the cloud servers you want to add, and click **OK**.

NOTE

To ensure a successful health check and service availability, configure the security groups of the cloud servers to allow access from 100.125.0.0/16.

Step 8 Click **Finish**.

----End

Creating an ELB Channel

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **VPC Channels**.

Step 4 Click **Create ELB Channel**.

Step 5 Enter a channel name, select a load balancer, and click **OK**.

NOTE

- Each load balancer can be used to create only one ELB channel. ELB channels cannot be updated once created.
- ELB channels can be created only using private network load balancers. For more information, see .

----End

Follow-Up Operations

Create an API for backend services deployed in a VPC to balance loads.

3.7.2 Deleting a VPC Channel

Scenario

You can delete VPC channels you no longer require.

NOTE

VPC channels that are currently in use by published APIs cannot be deleted.

Prerequisites

You have created a VPC channel.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **VPC Channels**.

Step 4 Delete a VPC channel. You can use one of the following methods:

- In the **Operation** column of the VPC channel you want to delete, click **Delete**.
- Click the name of the target VPC channel, and click **Delete** in the upper right corner of the displayed VPC channel details page.

Step 5 Click **Yes**.

----End

3.7.3 Editing Health Check Configurations

Scenario

You can modify the health check configurations of a VPC channel to meet service requirements.

Prerequisites

You have created a VPC channel.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **VPC Channels**.

Step 4 Click the name of the target VPC channel.

Step 5 Click the **Health Check** tab.

Step 6 Click **Edit Health Check**.

Step 7 In the **Edit Health Check Configuration** dialog box, modify the parameters listed in **Table 3-25**.

Table 3-25 Health check configurations

Parameter	Description
Protocol	The protocol used to perform health checks on cloud servers associated with the VPC channel. Options: • TCP • HTTP • HTTPS Default value: **TCP**.
Path	The destination path for health checks. Set this parameter only when **Protocol** is not set to **TCP**.
Check Port	The destination port for health checks. By default, the port of the VPC channel will be used.
Healthy Threshold	The number of consecutive successful checks required for a cloud server to be considered healthy. Range: 2–10. Default value: **2**.
Unhealthy Threshold	The number of consecutive failed checks required for a cloud server to be considered unhealthy. Range: 2–10. Default value: **5**.
Timeout (s)	The timeout used to determine whether a health check has failed. Unit: s. Range: 2–30. Default value: **5**.
Interval (s)	The interval between consecutive checks. Unit: s. Range: 5–300. Default value: **10**.
Response Codes	The HTTP codes used to check for a successful response from a target. Set this parameter only when **Protocol** is not set to **TCP**.

Step 8 Click **OK**.

----End

3.7.4 Editing Cloud Server Configurations of a VPC Channel

Scenario

You can add or remove cloud servers and edit cloud server weights for VPC channels to meet service requirements.

Prerequisites

You have created a VPC channel.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **VPC Channels**.

Step 4 Click the name of the target VPC channel.

Step 5 Click the **Cloud Servers** tab.

Step 6 Add or remove cloud servers and edit cloud server weights.

- Adding cloud servers
  a. Click **Select Cloud Server**.
  b. Select the cloud servers you want to add, set cloud server weights, and click **OK**.

  NOTE

  To ensure a successful health check and service availability, configure the security groups of the backend cloud servers to allow access from 100.125.0.0/16.
- Removing cloud servers
  a. In the **Operation** column of the cloud servers you want to remove, click **Remove**.
  b. Click **Yes**.
- Editing the weight of a cloud server
  a. In the **Weight** column of the target cloud server, click  .
  b. Change the weight and click .
- Editing the weights of multiple cloud servers
  a. Select the cloud servers to be edited, and click **Edit Weight**.
  b. Change the weights of the selected cloud servers, and click **OK**.

----End

3.8 Custom Authorizers

3.8.1 Creating a Custom Authorizer

Scenario

API Gateway supports custom authentication of both frontend and backend requests.

- Frontend custom authentication: If you already have an authentication system, you can configure it in a function and then create a custom authorizer by using the function to authenticate API requests.

- Backend custom authentication: You can create a custom authorizer to authenticate requests for different backend services, eliminating the need to customize APIs for different authentication systems and simplifying API development. You only need to create a function-based custom authorizer in API Gateway to connect to the backend authentication system.

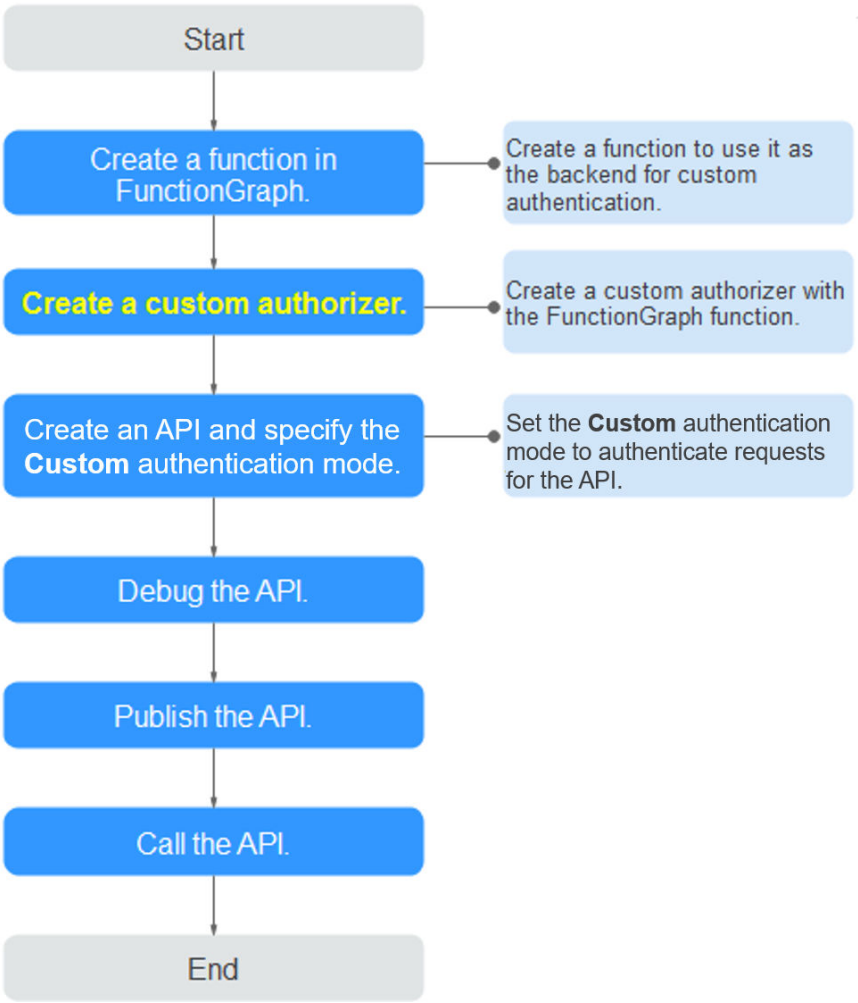
NOTE

Custom authentication is implemented using FunctionGraph and not supported if FunctionGraph is unavailable in the selected region.

For details about custom authentication, see *Developer Guide*.

The following figure shows the process of calling APIs through custom authentication.

Figure 3-6 Calling APIs through custom authentication



Prerequisites

You have created a function in FunctionGraph.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 Choose **API Publishing** > **Custom Authorizers**, and click **Create Custom Authorizer**.

Step 4 Set the parameters listed in **Table 3-26**.

Table 3-26 Parameters for creating a custom authorizer

Parameter	Description
Name	Authorizer name.
Type	• **Frontend**: Authenticates access to APIs. • **Backend**: Authenticates access to backend services.
Function URN	Select a FunctionGraph function.
Identity Sources	Request parameters used for authentication. This parameter is mandatory only if you set **Type** to **Frontend**, and **Max. Cache Age (s)** is greater than **0**. When the cache is used, this parameter is used as a search criterion to query authentication results.
Max. Cache Age (s)	The time for caching authentication results. Value **0** means that authentication results will not be cached. The maximum value is **3600**.
User Data	Customized request parameters to be used together with **Identity Sources** when API Gateway invokes a function.

Step 5 Click **OK**.

----End

3.8.2 Deleting a Custom Authorizer

Scenario

You can delete custom authorizers you no longer require.

NOTE

- Custom authentication is implemented using FunctionGraph and not supported if FunctionGraph is unavailable in the selected region.
- Custom authorizers that have been configured for APIs cannot be deleted.

Prerequisites

You have **created a custom authorizer**.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 Choose **API Publishing** > **Custom Authorizers**, and click **Delete** in the row containing the custom authorizer you want to delete.

Step 4 Click **Yes**.

----End

3.9 Monitoring

3.9.1 API Gateway Metrics

Introduction

This section describes the metrics that API Gateway reports to the Cloud Eye service. You can view metrics and alarms by using the Cloud Eye console.

Namespace

Shared gateway: SYS.APIG

Metrics

Table 3-27 Shared gateway metrics

Metric ID	Metric Name	Description	Value Range	Monitored Object and Dimension	Monitoring Interval (Minute)
avg_latency	Average Latency	Average latency of the API.	≥ 0 Unit: ms	Monitored object: an API Dimension: api_id	1
input_throughput	Incoming Traffic	Incoming traffic of the API.	≥ 0 Unit: Byte, KB, MB, or GB	Monitored object: an API Dimension: api_id	1

Metric ID	Metric Name	Description	Value Range	Monitored Object and Dimension	Monitoring Interval (Minute)
max_latency	Maximum Latency	Maximum latency of the API.	≥ 0 Unit: ms	Monitored object: an API Dimension: api_id	1
output_throughput	Outgoing Traffic	Outgoing traffic of the API.	≥ 0 Unit: Byte, KB, MB, or GB	Monitored object: an API Dimension: api_id	1
req_count	Requests	Number of times that the API has been called.	≥ 0	Monitored object: an API Dimension: api_id	1
req_count_2xx	2xx Responses	Number of times that the API returns a 2xx response.	≥ 0	Monitored object: an API Dimension: api_id	1
req_count_4xx	4xx Errors	Number of times that the API returns a 4xx error.	≥ 0	Monitored object: an API Dimension: api_id	1
req_count_5xx	5xx Errors	Number of times that the API returns a 5xx error.	≥ 0	Monitored object: an API Dimension: api_id	1
req_count_error	Total Errors	Total number of errors returned by the API.	≥ 0	Monitored object: an API Dimension: api_id	1

Dimension

Table 3-28 Shared gateway monitoring dimension

Key	Value
api_id	API

3.9.2 Creating Alarm Rules

Scenario

You can create alarm rules to monitor the status of your APIs.

An alarm rule consists of a rule name, monitored objects, metrics, alarm thresholds, monitoring interval, and notification.

Prerequisites

An API has been called.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **APIs**.

Step 4 Click the name of the target API.

Step 5 On the **Dashboard** tab page, click **View Metric** to access the Cloud Eye console. Then create an alarm rule. For details, see "Creating an Alarm Rule" of the *Cloud Eye User Guide*.

----End

3.9.3 Viewing Metrics

Scenario

Cloud Eye monitors the status of your APIs and allows you to view their metrics.

Prerequisites

You have created an API group and API.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Publishing** > **APIs**.

Step 4 Click the name of the target API.

API metrics are displayed on the **Dashboard** tab page.

Step 5 Click **View Metric** to view more metrics on the Cloud Eye console.

NOTE

The monitoring data is retained for two days. To retain the data for a longer period, save it to an OBS bucket.

----End

4 API Calling

4.1 App Management

4.1.1 Creating an App and Obtaining Authorization

Scenario

For an API that uses app authentication, you can create an app and use the app and its ID and key pair (AppKey and AppSecret) to call the API. You can use an app to call an API only after you bind the app to the API. When you call the API, replace the key pair in the SDK with your own key pair so that API Gateway can authenticate your identity. For details about app authentication, see *Developer Guide*.

NOTE

- If the authentication mode of the target API has been set to **None** or **IAM**, you do not need to create apps to call this API.
- You can create a maximum of 50 apps.

Creating an App

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Calling** > **Apps**.

Step 4 Click **Create App**, and set the parameters listed in **Table 4-1**.

Table 4-1 App information

Parameter	Description
Name	App name.
Description	Description of the app.

Step 5 Click **OK**.

After the app is created, its name and ID are displayed in the app list.

Step 6 Click the app name, and view the AppKey and AppSecret on the app details page.

----End

Binding an App to an API

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Calling** > **Apps**.

Step 4 Bind an app to an API. You can use one of the following methods:

- In the **Operation** column of the app, click **Bind to API**, and then click **Select API**.
- Click the name of the target app, and click **Select API**.

Step 5 Select an environment, select an API, and click **OK**.

After the binding is complete, you can view the API on the app details page.

NOTE

- If you do not intend to use the app to access an API, click **Unbind** in the **Operation** column of the API to remove the API.
- To debug an API to which the app is bound, click **Debug** in the row containing the API.

----End

Follow-Up Operations

You can **call APIs** through different authentication modes.

4.1.2 Deleting an App

Scenario

You can delete apps you no longer require.

Prerequisites

You have created an app.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Calling** > **Apps**.

Step 4 Delete an app. You can use one of the following methods:

- In the **Operation** column of the app you want to delete, click **Delete**.
- Click the name of the target app, and click **Delete App** in the upper right corner of the displayed app details page.

NOTE

If the app has been bound to any APIs, you must unbind the app and then delete it.

Step 5 Click **Yes**.

----End

4.1.3 Resetting the AppSecret of an App

Scenario

You can reset the AppSecret of an app after the app is created.

However, when you reset the AppSecret, it becomes invalid and APIs bound to the app cannot be called. To enable API calls for that app again, you will need to update the AppSecret of the app you use.

Prerequisites

You have created an app.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Calling** > **Apps**.

Step 4 Click the name of the target app.

Step 5 In the upper right corner of the displayed app details page, click **Reset AppSecret**.

Step 6 Click **Yes**.

----End

4.1.4 Adding an AppCode for Simple Authentication

Scenario

AppCodes are identity credentials of an app used to call APIs in simple authentication mode. In this mode, the **X-Apig-AppCode** parameter (whose value is an AppCode on the app details page) is added to the HTTP request header for quick response. API Gateway verifies only the AppCode and the request content does not need to be signed.

When an API is called using app authentication and simple authentication is enabled for the API, AppKey and AppSecret can be used to sign and verify the API request. AppCode can also be used for simple authentication.

NOTE

- For security purposes, simple authentication only supports API calls over HTTPS.
- You can create a maximum of five AppCodes for each app.

Prerequisites

You have created an app.

Generating an AppCode

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Calling** > **Apps**.

Step 4 Click the name of the target app.

Step 5 Click the **AppCodes** tab.

Step 6 Click **Add AppCode** to generate an AppCode. It can be automatically generated or customized.

----End

Using AppCode for Simple Authentication of API Requests

Step 1 When creating an API, set **Security Authentication** to **App** and enable **Simple Authentication**.

NOTE

After you enable simple authentication for an existing API, you need to publish the API again to make the configuration take effect.

Step 2 Bind an app to the API.

Step 3 When sending a request, add the **X-Apig-AppCode** parameter to the request header and omit the request signature.

For example, when using curl, add the **X-Apig-AppCode** parameter to the request header and set the parameter value to the **generated AppCode**.

```
curl -X GET "https://api.exampledemo.com/testapi" -H "content-type: application/json"  -H "host: api.exampledemo.com" -H "X-Apig-AppCode: xhrJVJKABSOxc7d**********FZL4gSHEXkCMQC"
```

----End

4.1.5 Viewing API Details

Scenario

You can view the details of an API to which an app has been bound.

Prerequisites

- You have created an app.
- The app has been bound to an API.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Calling** > **Apps**.

Step 4 Click the name of the target app.

Step 5 Click the name of the target API to view its details.

----End

4.2 SDKs

API Gateway supports API authentication based on IAM, apps, and custom authorizers. You can also choose not to authenticate API requests. For details about the differences between the authentication modes, see "How Do I Choose an Authentication Mode" in the *API Gateway Developer Guide*.

This section describes how to download SDKs and view related instructions.

For details about IAM authentication, see "Calling APIs Through IAM Authentication" in the *Developer Guide*.

Scenario

SDKs are used when you call APIs through app authentication. Download SDKs and related documentation and then call APIs by following the instructions in the documentation.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner and choose **API Gateway**.

Step 3 In the navigation pane, choose **API Calling** > **SDKs**.

Step 4 Click **Download SDK** of the desired language.

To view the support guide, click **SDK Documentation**.

----End

4.3 Calling Published APIs

4.3.1 Calling APIs

Obtaining APIs and Documentation

Before calling APIs, obtain the request information from the API provider, including the access domain name, protocol, method, path, and request parameters.

Obtain APIs: from your company or from a partner

Obtain related documentation from the help center of the API provider's official website:

The authentication information to be obtained varies with the API authentication mode.

- App authentication:
  - Signature authentication: Obtain the key and secret (or client AppKey and AppSecret) of the app authorized for the API from the API provider as well as the SDK for calling the API.
  - Simple authentication: Obtain the AppCode of the app authorized for the API from the API provider.
  - Other authentication modes: Obtain the key and secret (or client AppKey and AppSecret) of the app authorized for the API from the API provider.
- IAM authentication: The account credential (token or AK/SK obtained with the account and password) obtained on the cloud service platform is used for authentication. If the AK/SK is used for authentication, you also need to obtain the SDK from the API provider for calling the API.
- Custom authentication: Obtain the custom authentication information to be carried in the request parameters from the API provider.
- None: No authentication information is required.

Calling an API

Step 1 Set the request path.

Scenario	Request Parameter Configuration
Calling an API with a domain name	Call the API using the subdomain name allocated to the API group or a domain name bound to the group. No additional configuration is required.

Step 2 Set the authentication parameters.

Authentication Mode	Request Parameter Configuration
App authentication (with a signature)	Use the SDK to sign API requests. For details, see section "Calling APIs Through App Authentication" in the *Developer Guide*.
App authentication (through simple authentication)	Add the header parameter **X-Apig-AppCode** and set the parameter value to the AppCode obtained in **Obtaining APIs and Documentation**.
IAM authentication (with a token)	Obtain a token from the cloud platform and carry the token in API requests for authentication. For details, see section "Token Authentication" in the *Developer Guide*.
IAM authentication (with AK/SK)	Use an SDK to sign API requests. For details, see section "AK/SK Authentication" in the *Developer Guide*.
Custom authentication	Carry authentication information in API request parameters for authentication.
None	Call APIs without authentication.

----End

4.3.2 Response Headers

The following table describes the response headers that API Gateway adds to the response returned when an API is called.

X-Apig-Mode: debug indicates API debugging information.

Response Header	Description	Remarks
X-Request-Id	Request ID.	Returned for all valid requests.
X-Apig-Latency	Duration from the time when API Gateway receives a request to the time when the backend returns a message header.	Returned only when the request header contains **X-Apig-Mode: debug**.

Response Header	Description	Remarks
X-Apig-Upstream-Latency	Duration from the time when API Gateway sends a request to the backend to the time when the backend returns a message header.	Returned only when the request header contains **X-Apig-Mode: debug** and the backend type is not Mock.
X-Apig-RateLimit-api	API request limit information. Example: **remain:9,limit:10,time:10 second**.	Returned only when the request header contains **X-Apig-Mode: debug** and a limit has been configured for the number of times the API can be called.
X-Apig-RateLimit-user	User request limit information. Example: **remain:9,limit:10,time:10 second**.	Returned only when the request header contains **X-Apig-Mode: debug** and a limit has been configured for the number of times the API can be called by a user.
X-Apig-RateLimit-app	App request limit information. Example: **remain:9,limit:10,time:10 second**.	Returned only when the request header contains **X-Apig-Mode: debug** and a limit has been configured for the number of times the API can be called by an app.
X-Apig-RateLimit-ip	IP address request limit information. Example: **remain:9,limit:10,time:10 second**.	Returned only when the request header contains **X-Apig-Mode: debug** and a limit has been configured for the number of times the API can be called by an IP address.
X-Apig-RateLimit-api-allenv	Default API request limit information. Example: **remain:199,limit:200,time:1 second**.	Returned only when the request header contains **X-Apig-Mode: debug**.

4.3.3 Error Codes

Table 4-2 lists the error codes that you may encounter when calling APIs.

NOTE

- For details about the error codes that may occur when you manage APIs, see .
- If an error occurs when you use API Gateway, find the error message and description in the following table according to the error code, for example, APIG.0101. The error messages are subject to change without prior notice.

Table 4-2 Error codes

Error Code	Error Message	HTTP Status Code	Description	Solution
APIG.0101	The API does not exist or has not been published in the environment.	404	The API does not exist or has not been published in the environment.	Check whether the domain name, method, and path are consistent with those of the registered API. Check whether the API has been published. If it has been published in a non-production environment, check whether the X-Stage header in the request is the environment name. Check whether the domain name used to call the API has been bound to the group to which the API belongs.
APIG.0101	The API does not exist.	404	The API request method does not exist.	Check whether the API request method is the same as the method defined by the API.
APIG.0103	The backend does not exist.	500	The backend service was not found.	Contact technical support.
APIG.0104	The plug-ins do not exist.	500	No plug-in configurations were found.	Contact technical support.
APIG.0105	The backend configurations do not exist.	500	No backend configurations were found.	Contact technical support.

Error Code	Error Message	HTTP Status Code	Description	Solution
APIG.0106	Orchestration error.	400	An orchestration error occurred.	Check whether the frontend and backend parameters of the API are correct.
APIG.0201	API request error.	400	Invalid request parameters.	Set valid request parameters.
APIG.0201	Request entity too large.	413	The request body exceeds 12 MB.	Reduce the size of the request body.
APIG.0201	Request URI too large.	414	The request URI exceeds 32 KB.	Reduce the size of the request URI.
APIG.0201	Request headers too large.	494	The request headers are too large because one of them exceeds 32 KB or the total length exceeds 128 KB.	Reduce the size of the request headers.
APIG.0201	Backend unavailable.	502	The backend service is unavailable.	Check whether the backend address configured for the API is accessible.
APIG.0201	Backend timeout.	504	The backend service has timed out.	Increase the timeout duration of the backend service or shorten the processing time.
APIG.0201	An unexpected error occurred	500	An internal error occurred.	Contact technical support.
APIG.0204	SSL protocol is not supported: TLSv1.1	400	The SSL protocol version is not supported.	Use a supported SSL protocol version.
APIG.0301	Incorrect IAM authentication information.	401	The IAM authentication details are incorrect.	Check whether the token is correct.

Error Code	Error Message	HTTP Status Code	Description	Solution
APIG.0302	The IAM user is not authorized to access the API.	403	The IAM user is not allowed to access the API.	Check whether the user is controlled by a blacklist or whitelist.
APIG.0303	Incorrect app authentication information.	401	The app authentication details are incorrect.	Check whether the request method, path, query strings, and request body are consistent with those used for signing; check whether the date and time on the client are correct; and check whether the signing code is correct by referring to .
APIG.0304	The app is not authorized to access the API.	403	The app is not allowed to access the API.	Check whether the app has been authorized to access the API.
APIG.0305	Incorrect authentication information.	401	The authentication information is incorrect.	Check whether the authentication information is correct.
APIG.0306	API access denied.	403	Access to the API is not allowed.	Check whether you have been authorized to access the API.
APIG.0307	The token must be updated.	401	The token needs to be updated.	Obtain a new token from IAM.

Error Code	Error Message	HTTP Status Code	Description	Solution
APIG.0308	The throttling threshold has been reached.	429	The throttling threshold has been reached.	Try again after the throttling resumes. If the number of subdomain requests per day is reached, bind an independent domain name to the API.
APIG.0310	The project is unavailable.	403	The project is currently unavailable.	Select another project and try again.
APIG.0311	Incorrect debugging authentication information.	401	The debugging authentication details are incorrect.	Contact technical support.
APIG.0401	Unknown client IP address.	403	The client IP address cannot be identified.	Contact technical support.
APIG.0402	The IP address is not authorized to access the API.	403	The IP address is not allowed to access the API.	Check whether the IP address is controlled by a blacklist or whitelist.
APIG.0404	Access to the backend IP address has been denied.	403	The backend IP address cannot be accessed.	Check whether the backend IP address or the IP address corresponding to the backend domain name is accessible.
APIG.0502	The app has been frozen.	405	The app has been frozen.	Check whether your account balance is sufficient.
APIG.0601	Internal server error.	500	An internal error occurred.	Contact technical support.
APIG.0602	Bad request.	400	Invalid request.	Check whether the request is valid.

Error Code	Error Message	HTTP Status Code	Description	Solution
APIG.0605	Domain name resolution failed.	500	Domain name resolution failed.	Check whether the domain name is correct and has been bound to a correct backend address.
APIG.0606	Failed to load the API configurations.	500	API configurations could not be loaded.	Contact technical support.
APIG.0607	The following protocol is supported: {xxx}	400	The protocol is not supported. Only *xxx* is supported. *xxx* is subject to the actual value in the response.	Use HTTP or HTTPS to access the API.
APIG.0608	Failed to obtain the admin token.	500	The administrator account details cannot be obtained.	Contact technical support.
APIG.0609	The VPC backend does not exist.	500	The VPC backend service cannot be found.	Contact technical support.
APIG.0610	No backend available.	502	No backend services are available.	Check whether all backend services are available. For example, check whether the API calling information is consistent with the actual configuration.
APIG.0611	The backend port does not exist.	500	The backend port was not found.	Contact technical support.

Error Code	Error Message	HTTP Status Code	Description	Solution
APIG.0612	An API cannot call itself.	500	An API cannot call itself.	Modify the backend configurations, and ensure that the number of layers the API is recursively called does not exceed 10.
APIG.0613	The IAM service is currently unavailable.	503	IAM is currently unavailable.	Contact technical support.
APIG.0705	Backend signature calculation failed.	500	Backend signature calculation failed.	Contact technical support.
APIG.0802	The IAM user is forbidden in the currently selected region	403	The IAM user is disabled in the current region.	Contact technical support.
APIG.1009	AppKey or AppSecret is invalid	400	The AppKey or AppSecret is invalid.	Check whether the AppKey or AppSecret in the request is correct.

5 Auditing

Enabling CTS

If you want to collect, record, or query operation logs for API Gateway in common scenarios such as security analysis, compliance audit, resource tracing, and problem locating, enable Cloud Trace Service (CTS). For details, see chapter "Enabling CTS" in the *Cloud Trace Service User Guide*.

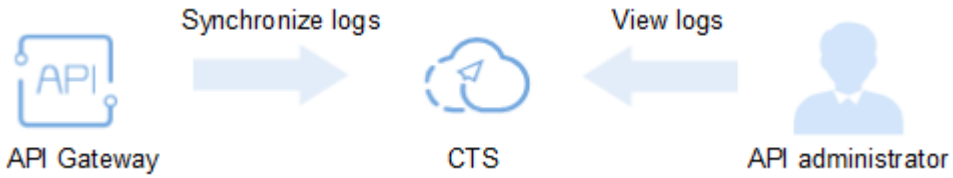
CTS provides the following functions:

- Recording audit logs
- Querying audit logs
- Dumping audit logs
- Encrypting trace files
- Enabling notifications of key operations

Querying Audit Logs

Query audit logs by following the procedure in chapter "Querying Real-Time Traces" in the *Cloud Trace Service User Guide*.

Figure 5-1 Viewing logs



Viewing Key Operations

With CTS, you can record operations associated with API Gateway for future query, audit, and backtracking.

Disabling CTS

Disable CTS by following the procedure in chapter "Deleting a Tracker" in the *Cloud Trace Service User Guide*.

6 Quota Management

To change the default quota limits and increase the quotas, see **Help Center > Others > FAQs > How Do I Apply for a Higher Quota?**

Table 6-1 API Gateway quotas

Item	Default Restriction	Changeable
Number of API groups	50	√
Number of APIs	200	√
Number of backend policies	5	√
Number of apps	You can create a maximum of 50 apps. The app quota includes created apps and apps generated when APIs are purchased from the API Marketplace.	√
Number of request throttling policies	• You can create a maximum of 30 request throttling policies. • The call limit for a single user cannot exceed that for the target API. • The call limit for a single app cannot exceed that for a single user. • The call limit for a single IP address cannot exceed that for the target API.	√
Number of environments	10	√
Number of signature keys	30	√

Item	Default Restriction	Changeable
Number of access control policies	100	√
Number of VPC channels	30	√
Number of variables	You can create a maximum of 50 variables for an API group in each environment.	√
Number of independent domain names	A maximum of five independent domain names can be bound to an API group.	√
Cloud servers	A maximum of 200 cloud servers can be added to a VPC channel.	√
Number of parameters	A maximum of 50 parameters can be created for an API.	√
Number of API publication records	A maximum of 10 publication records of an API can be retained for each environment.	√
API access rate	Up to 200 times per second	√
Excluded apps	A maximum of 30 excluded apps can be added to a request throttling policy	√
Excluded tenants	A maximum of 30 excluded tenants can be added to a request throttling policy	√
Access to a subdomain name	A subdomain name can be accessed up to 1000 times a day.	x
Maximum size of an API request package	12 MB	x
TLS protocol	TLS 1.1 and TLS 1.2 are supported. TLS 1.2 is recommended.	x
Real-name authentication	Users who have not completed real-name authentication cannot create any resources.	x