API Gateway

User Guide (API Publishing)

Issue 16
Date 2019-06-28
Copyright © Huawei Technologies Co., Ltd. 2019. All rights reserved.
No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

Trademarks and Permissions

and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd. All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.
# Contents

1 Overview......................................................................................................................................... 1

2 Permissions Management............................................................................................................2
2.1 Basic Concepts............................................................................................................................................................... 2
2.2 Creating a User and Granting Permissions.....................................................................................................................7
2.3 Policy Syntax................................................................................................................................................................13

3 Experiencing Demo..................................................................................................................... 15

4 Getting Started............................................................................................................................. 17
4.1 Introduction.................................................................................................................................................................. 17
4.2 Creating an API Group................................................................................................................................................. 18
4.3 Binding a Domain Name.............................................................................................................................................. 19
4.4 Creating an API............................................................................................................................................................ 21
4.5 Debugging an API........................................................................................................................................................ 32
4.6 (Optional) Creating an Environment and Environment Variable................................................................................. 34
4.7 Publishing an API.........................................................................................................................................................37

5 API Group Management............................................................................................................ 40
5.1 Creating an API Group................................................................................................................................................. 40
5.2 Binding a Domain Name.............................................................................................................................................. 42
5.3 Deleting an API Group................................................................................................................................................. 43

6 API Management......................................................................................................................... 45
6.1 Creating an API............................................................................................................................................................ 45
6.2 Enabling CORS............................................................................................................................................................ 56
6.3 Debugging an API........................................................................................................................................................ 60
6.4 Authorizing Apps to Call an API................................................................................................................................. 62
6.5 Publishing an API.........................................................................................................................................................64
6.6 Taking an API Offline...................................................................................................................................................66
6.7 Deleting an API............................................................................................................................................................ 67
6.8 Importing APIs............................................................................................................................................................. 68
6.9 Exporting APIs............................................................................................................................................................. 70

7 Request Throttling...................................................................................................................... 72
7.1 Creating a Request Throttling Policy..........................................................................................................................72
7.2 Deleting a Request Throttling Policy............................................................................................................................75
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.3 Adding an Excluded App or Tenant</td>
<td>76</td>
</tr>
<tr>
<td>7.4 Removing an Excluded App or Tenant</td>
<td>78</td>
</tr>
<tr>
<td>8 Access Control</td>
<td>80</td>
</tr>
<tr>
<td>8.1 Creating an Access Control Policy</td>
<td>80</td>
</tr>
<tr>
<td>8.2 Deleting an Access Control Policy</td>
<td>82</td>
</tr>
<tr>
<td>9 Environment Management</td>
<td>84</td>
</tr>
<tr>
<td>9.1 Creating Environments and Environment Variables</td>
<td>84</td>
</tr>
<tr>
<td>9.2 Deleting an Environment</td>
<td>87</td>
</tr>
<tr>
<td>10 Signature Key Management</td>
<td>88</td>
</tr>
<tr>
<td>10.1 Creating and Using a Signature Key</td>
<td>88</td>
</tr>
<tr>
<td>10.2 Deleting a Signature Key</td>
<td>90</td>
</tr>
<tr>
<td>11 VPC Channel Management</td>
<td>92</td>
</tr>
<tr>
<td>11.1 Creating a VPC Channel</td>
<td>92</td>
</tr>
<tr>
<td>11.2 Deleting a VPC Channel</td>
<td>95</td>
</tr>
<tr>
<td>11.3 Editing Health Check Configurations</td>
<td>96</td>
</tr>
<tr>
<td>11.4 Editing ECS Configurations of a VPC Channel</td>
<td>97</td>
</tr>
<tr>
<td>12 Auditing</td>
<td>99</td>
</tr>
<tr>
<td>13 Monitoring</td>
<td>100</td>
</tr>
<tr>
<td>13.1 API Gateway Metrics</td>
<td>100</td>
</tr>
<tr>
<td>13.2 Creating Alarm Rules</td>
<td>100</td>
</tr>
<tr>
<td>13.3 Viewing Metrics</td>
<td>101</td>
</tr>
<tr>
<td>A Change History</td>
<td>102</td>
</tr>
</tbody>
</table>
1 Overview

API Gateway is a high-performance, high-availability, and high-security API hosting service that helps enterprises build, manage, and deploy Application Programming Interfaces (APIs) at any scale. With just a few clicks, you can implement system integration and open and monetize well-developed service capabilities while minimizing costs and risks.

Figure 1-1 API Gateway architecture

This document provides guidance for enterprises and developers on opening up services and data through API Gateway to monetize service capabilities.

To obtain and invoke APIs of other providers through the marketplace or other channels, see the User Guide (API Calling).

Using API Gateway, you can:

- Quickly create APIs on a visualized console.
- Easily debug APIs with an inline debugging tool.
- Publish APIs in specific environments for intended users to invoke.
- Limit the maximum number of times each API can be called within a period.
- Control API access from specific sources by defining a blacklist or whitelist.
- Create signature keys for APIs to ensure secure access to backend services.
- Balance the loads of backend servers by configuring Virtual Private Cloud (VPC) channels.
2 Permissions Management

2.1 Basic Concepts

This chapter describes Identity and Access Management (IAM)'s fine-grained permissions management for your API Gateway. With IAM, you can:

- Create IAM users for employees based on the organizational structure of your enterprise. Each IAM user has their own security credentials, providing access to API Gateway resources.
- Grant only the permissions required for users to perform a task.
- Entrust a HUAWEI CLOUD account or cloud service to perform professional and efficient O&M on your API Gateway resources.

If your HUAWEI CLOUD account does not need individual IAM users, then you may skip over this topic.

The following sections describe the common IAM operations, including creating a user and user group, granting permissions to a user group, and creating a custom policy. For other IAM operations, see the IAM User Guide.

Account

To use HUAWEI CLOUD, you need to register an account using your mobile number. The account owns your HUAWEI CLOUD resources and has full access permissions for the resources. You can use the account to reset user passwords and assign permissions. Your account receives and pays all bills generated by your IAM users' use of resources. To log in to the HUAWEI CLOUD management console using an account, choose Account Login.

If you forget the password of the account, you can reset it by following the procedure in Resetting Password for a HUAWEI CLOUD Account.
IAM User

IAM users are created by an account to use cloud services. Each IAM user has their own password and access key to access HUAWEI CLOUD using the console or APIs. The users manage cloud resources for the account based on assigned permissions. IAM users do not own resources or make payments. It is the account that controls user permissions and pays the bills. To log in to the HUAWEI CLOUD management console as an IAM user, choose IAM User Login.

If you forget the password of an IAM user, you can reset it by following the procedure in Resetting Password for an IAM User.
Relationship Between an Account and Its IAM Users

An account and its IAM users are like a parent and children. The account owns the resources and makes payments for IAM users' resource usage. It has full access permissions for these resources. IAM users are created by the account, and only have the permissions granted by the account. The account can modify or cancel the IAM users' permissions at any time. Fees generated by IAM users' use of resources are paid by the account.
Identity Credentials

Identity credentials are used for authentication when you or your IAM users access HUAWEI CLOUD through the console or APIs. Identity credentials include the password and access keys, which can be managed in IAM.

- Password: A common identity credential for logging in to the HUAWEI CLOUD management console or calling HUAWEI CLOUD APIs.
- Access key: An access key ID/secret access key (AK/SK) pair, which is used only for calling HUAWEI CLOUD APIs. Each access key provides a signature for cryptographic authentication to ensure that access requests are secret, complete, and correct.

User Group

User groups facilitate centralized user management and streamlined permissions management. Users in the same user group have the same permissions. IAM users must be added to a user group to obtain the permissions required for accessing specified resources or cloud services in the account. A user can be added to multiple groups, which allows them to inherit different permissions.

The default user group admin has all of the permissions required to use all of the cloud resources. Users in this group can perform operations on all the resources, including but not limited to creating user groups and users, assigning permissions, and managing resources.
Authorization

Authorization is the process of granting required permissions for a user to perform a task. After a system or custom policy is assigned to a user group, users in the group inherit the permissions defined by the policy to manage resources. For example, managing ECSs.

Project

A project corresponds to a HUAWEI CLOUD region. Default projects are defined to group and physically isolate resources (including computing, storage, and network resources) across regions. Users can be granted permissions in a default project to access all resources in the region associated with the project. If you need more refined access control, you can create subprojects under a default project and purchase resources in subprojects. Then you can assign required permissions for users to access only the resources in specific subprojects.
2.2 Creating a User and Granting Permissions

This topic describes how to use a group to grant permissions to a user. **Figure 2-7** shows the process for granting permissions.

![Figure 2-7 Process for granting API Gateway permissions](image)

1. Create a user group and grant permissions to it.
   
   Create a user group on the IAM console, and assign the **APIG Administrator** policy to the group.

2. Create a user.
   
   Create a user on the IAM console and add the user to the group created in 1.

3. Log in and verify permissions.
   
   Log in to the API Gateway console as the created user, and verify that it has the administrator permissions for API Gateway.

**Prerequisites**

Before assigning permissions to user groups, you should learn about the system policies listed in **Table 2-1**. For the system policies of other services, see **Permission Description**.
### Table 2-1 System policy summary

<table>
<thead>
<tr>
<th>Policy Name</th>
<th>Description</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>APIG Adminstrator</td>
<td>Administrator permissions for API Gateway</td>
<td>Creating, publishing, and deleting APIs</td>
</tr>
</tbody>
</table>

### Step 1: Create a User Group and Grant Permissions

User groups facilitate centralized user management and streamlined permissions management. Users in the same user group have the same permissions. Users created in IAM inherit permissions from the groups to which they belong. To create a user group and grant it permissions, perform the following steps:

**Step 1** Log in to HUAWEI CLOUD using an account.

![Account login](image)

**Step 2** On the management console, click the username on the upper right corner and then choose **Identity and Access Management**.
**Step 3**  On the IAM console, choose **User Groups** in the navigation pane. Then click **Create User Group**.

**Figure 2-10 Creating a user group**

**Step 4**  Enter a user group name, for example, **Developers**, and click **OK**.

The user group is displayed in the user group list.

**Step 5**  In the same row as the user group, choose **More > Configure Permission**. Then click **Configure Policy** next to the region for which you want to assign permissions to the user group.

API Gateway is a project-level service. Therefore, you need to assign permissions in the projects in which users in the group need to access API Gateway resources.

**Figure 2-11 Granting permissions**

**Figure 2-12 Selecting policies**
Step 6  In the **Configure Policy** dialog box, search for and select **APIG Administrator**.

**Figure 2-13** Selecting the APIG Administrator policy

Step 7  Click **OK**.

**Figure 2-14** Permissions granted

---End

**Step 2: Create an IAM User**

IAM users can be created for employees or applications of an enterprise. Each IAM user has their own security credentials, and inherits permissions from the groups it is a member of. To create an IAM user, perform the following steps:

**Step 1**  In the navigation pane of the IAM console, choose **Users**. Then click **Create User**.

**Step 2**  Set user information and click **Next**.
Username: Used for logging in to HUAWEI CLOUD. For this example, enter James.

Credential Type: Identity credential for authentication. For this example, select Password.

- Password: Used for accessing HUAWEI CLOUD using the console or development tools (including APIs, CLI, and SDKs).
- Access Key: Used for logging to HUAWEI CLOUD using development tools. This credential type is more secure, and is recommended if the user does not need to use the console.

(Optional) User Groups: Select Developers. The user will inherit the permissions granted to the user group. The default user group is admin, which has the administrator permissions and all of the permissions required to use all cloud resources.

(Optional) Description: Description of the user.

Step 3 On the next page, set required parameters, and click OK.

The following password types are available:

- Set at first login: Select this option if you are not the entity using user James. James will receive a one-time login URL by email and can set a password at first login.
- Automatically generated: Select this option if James accesses HUAWEI CLOUD using a development tool. HUAWEI CLOUD will generate a random 10-digit password.
- Set manually: Select this option if you are the entity using user James. Then set a password for login.

----End

Step 3: Log In and Verify Permissions

After the user is created, use the username and identity credential to log in to HUAWEI CLOUD, and verify that the user has the permissions defined by the APIG Administrator policy. For more login methods, see Signing In to HUAWEI CLOUD.

Step 1 On the HUAWEI CLOUD login page, click IAM User Login.
Step 2 Enter the account name, username, and password, and click Log In.

- The account name is the name of the HUAWEI CLOUD account that created the IAM user.
- The username and password are those set by the account when creating the IAM user.

If the login fails, contact the entity owning the account to verify the username and password. Alternatively, you can reset the password by following the procedure in Resetting Password for an IAM User.

Step 3 After successful login, switch to a region where the user has been granted permissions on the management console. The default region is CN North-Beijing1.

Step 4 Choose Service List > API Gateway. On the API Gateway console, choose API Publishing > API Groups, and then click Create API Group. If a message appears indicating
insufficient permissions to perform the operation, the permissions policy has already taken effect.

----End

2.3 Policy Syntax

The **APIG Administrator** policy is used as an example to describe the syntax of permissions policies.

```json
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Policy version</td>
<td>The value is fixed to <strong>1.0</strong>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Meaning</td>
<td>Value</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>Statement</td>
<td>Effect</td>
<td>Determines whether the operation defined in an action is allowed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Allow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Deny</td>
</tr>
</tbody>
</table>
| Action    | Operation to be performed on API Gateway | Format: Service name:Resource type:Operation.  
apig:*:*: Permissions for performing all operations on all resource types in API Gateway. |
3 Experiencing Demo

Scenarios

API Gateway provides a demo showing how to create and publish an API for looking up mobile number locations and bind a request throttling policy to the API.

During the demo, the system automatically performs the following operations:

- Creates an API group.
- Creates an API.
- Publishes the API in the RELEASE environment.
- Creates a request throttling policy.
- Binds the request throttling policy to the API.

The mobile number location lookup service has an independent API that can be called using the HTTP GET method to send mobile numbers to the backend. The backend then returns the mobile number location information.

**NOTE**

- To experience the demo again, first delete the API and API group created during the demo.
- If the request throttling policy created during the demo is not renamed or deleted, it will be used when you perform the demo again.

Procedure

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose **Service List** > **Application** > **API Gateway**.

**Step 4** In the navigation pane, choose **Dashboard**.

**Step 5** Click **Demo**.

**Step 6** Click **Next**.

**Step 7** Enter a mobile number in the Value column of the **Path Parameters** area, click **Send Request**, and then view the response.
If the debugging is successful, the HTTP status code **200** and mobile number location information are displayed.

If the debugging failed, an HTTP status code **4xx** or **5xx** is displayed. For details, see **Error Codes**.

**Figure 3-1** Debugging the API

**Step 8** Call the API.

1. Choose **API Publishing > APIs**, and click **QueryPhoneNumber**.

2. On the **Monitoring** tab page, click ![copy icon] to copy the URL.

**Figure 3-2** API details page

**APIS > QueryPhoneNumber**

API URL: [http://f19377719b49439b028fd6c49ab627.apigw.southchina.huawei.com/QueryPhoneNumber](http://f19377719b49439b028fd6c49ab627.apigw.southchina.huawei.com/QueryPhoneNumber)

3. Paste the URL in the address bar of a browser, replace `{phoneNumber}` with the mobile number you want to look up, and press **Enter** to view the location information of the mobile number.

**Figure 3-3** Calling result

```json
{
  "http": "http",
  "mobile": "13012345678",
  "province": "CNN",
  "city": "CNN",
  "zip": "10000",
  "scode": "50010",
  "spcode": "8100010",
  "message": "Success"
}
```

----End
4 Getting Started

4.1 Introduction

To open up an API or a group of APIs, you need to complete the following operations:

1. Creating an API Group
   An API group facilitates management of APIs used for the same service. Create an API group before creating APIs.

2. Binding a Domain Name
   Before opening an API, bind one or more independent domain names (custom domain names) to the group to which the API belongs. API callers will invoke the API by using the bound domain names.

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

4. Debugging an API
   You can verify the API service on the console by setting HTTP header and body parameters.

5. (Optional) Creating an Environment and Environment Variable
   The same API can be called in different scenarios, such as the production environment (RELEASE) and other custom environments. RELEASE is the default environment provided by API Gateway. Environment variables can be defined to allow the API to be called in different environments.

6. Publishing an API
   An API can be called only after it has been published in an environment. You can view the publication history (such as version, description, time, and environment) of the API, and roll back to different historical versions.

7. (Optional) List the API on the marketplace.
   If you want to sell the API on the marketplace, see How Do I List APIs on HUAWEI CLOUD Marketplace for Sale?
Ensure that the backend service is accessible.

By default, API Gateway sends the body of each request to the backend service. If the request body contains private or sensitive information, you are advised to provide an encryption mechanism to prevent information leakage.

By default, API Gateway sends the user IP address in the header of each request to the backend service. If the user IP address is private or sensitive information, you should provide a privacy statement for the API caller.

## 4.2 Creating an API Group

### Scenarios

Before creating an API, you must create an API group. An API group facilitates management of APIs used for the same service.

**NOTE**

- Each API belongs to only 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 to select a region.

**Step 3**  Choose **Service List > Application > API Gateway**.

**Step 4**  In the navigation pane, choose **API Publishing > API Groups**.

**Step 5**  Click **Create API Group**, and set the parameters listed in **Table 4-1** according to **Figure 4-2**.

![Figure 4-2 Creating an API group](image)
Table 4-1 Parameters for creating an API group

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>API group name</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the API group</td>
</tr>
</tbody>
</table>

Step 6  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 open APIs, you must bind independent domain names to the API group to make the APIs accessible to users.

--- End

Creating an API Group Using a REST API

Alternatively, create an API group by using a REST API provided by API Gateway. For more information, see the following reference.

Creating an API Group

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 invoke APIs in the group. For more information, see Binding a Domain Name.

4.3 Binding a Domain Name

Scenarios

Before an API is opened, one or more independent domain names must be bound 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 each API group.

Before binding a domain name, understand the following two concepts:

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

1. The independent domain name to be bound has been CNAMEd to the subdomain name of the API group. For details, see Adding a CNAME Record Set.
2. If the API group contains APIs that are called through HTTPS, SSL certificates need to be added to 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 to select a region.

Step 3 Choose Service List > Application > API Gateway.

Step 4 In the navigation pane, choose API Publishing > API Groups.

Step 5 Bind a domain name in either of the following two ways:

- 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 6 Click Bind Domain Name, enter a domain name, and click ✓. Figure 4-3 shows the successfully bound domain name.

![Figure 4-3 Successfully bound domain name](image)

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

Step 7 (Optional) If the API group contains APIs that are called through HTTPS, add SSL certificates to the group.

1. Click Add SSL Certificate.
2. Enter a certificate name, certificate content, and a key, and click OK.
To edit the 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 the SSL certificate is not needed, click Delete SSL Certificate in the same row to delete it.

---End

Binding a Domain Name Using a REST API

Alternatively, bind an independent domain name to an API group by using a REST API provided by API Gateway. For more information, see the following references.

Binding a Domain Name

Binding a Certificate

Follow-Up Operations

After binding independent domain names to an API group, create APIs in the group to open up your backend capabilities. For details, see Creating an API.

4.4 Creating an API

Scenarios

Create an API in API Gateway to open up your service capabilities.

To create an API, you need to set the basic information, and define API request, backend request, and responses.

NOTE

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

Prerequisites

- An API group has been created. If no API groups are available, click Create API Group to create one on the Set Basic Information page.
- If the backend service is deployed in a VPC, create a VPC channel for accessing the service by following the procedure in Creating a VPC Channel or by clicking Manage VPC Channel on the Define Backend Request page.

Setting Basic Information

Step 1 Log in to the management console.

Step 2 Click in the upper left corner to select a region.

Step 3 Choose Service List > Application > API Gateway.
Step 4 In the navigation pane, choose API Publishing > APIs.

Step 5 Click Create API, and set the parameters listed in Table 4-2 according to Figure 4-4.

Figure 4-4 Setting the basic information


Table 4-2 Basic information

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>API name</td>
</tr>
</tbody>
</table>
| API Group   | Group to which the API belongs
If no API groups are available, click Create API Group to create one. |
| Visibility  | Configure whether the API is available to the public. Options: 
- **Public**: The API can be released to the marketplace. 
- **Private**: The API is excluded when the API group to which it belongs is released to the marketplace. |
### Security Authentication

There are three authentication modes:
- **App**: API Gateway authenticates API requests.
- **IAM**: Identity and Access Management (IAM) authenticates API requests.
- **None**: No authentication is required.

The API calling method varies depending on the authentication mode. For details, see the Developer Guide.

**NOTICE**
- If the authentication mode of the API is set to **IAM**, any HUAWEI CLOUD tenant can access the API, which may result in excessive charges caused by malicious requests.
- If the authentication mode of the API is set to **None**, any public network user can access the API, which may also result in excessive charges caused by malicious requests.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain Name</td>
<td>Subdomain automatically allocated by the system to the API group</td>
</tr>
</tbody>
</table>

#### Step 6
Click Next.

#### End

### Defining API Request

**Step 1** On the **Define API Request** page, set parameters according to **Table 4-3**.

![Figure 4-5 Defining API request](image-url)

**Table 4-3 API request parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain Name</td>
<td>Subdomain automatically allocated by the system to the API group</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| Protocol  | Protocol used for invoking the API. Options:  
  - HTTP  
  - HTTPS  
  - HTTP&HTTPS  
  HTTPS is recommended for transmitting important or sensitive data. |
| Path      | Path for requesting the API  
  Enter a path in the format of `/serviceName/interfaceName`.  
  - The content in braces (`{}`) is case-sensitive.  
  - Parameters contained in the request path must be defined as input parameters. |
| Matching  | Options:  
  - **Exact match**: The API can be invoked using the specified request path.  
  - **Prefix match**: The API can be invoked using paths starting with the matching characters.  
    For example, if the request path is set to `/test/AA` and the matching mode to **Prefix match**, the API can be invoked using `/test/AA/CC` but cannot be invoked using `/test/AACC`.  
    **NOTE**  
    The plus (+) sign is not supported for prefix match. |
| Method    | API calling method. The options include **GET**, **POST**, **DELETE**, **PUT**, **PATCH**, **HEAD**, **OPTIONS**, and **ANY**.  
  **ANY** indicates that the API can be accessed using any request method. |
| CORS      | Enables or disables 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.  
  CORS requests are classified into two types:  
  - Simple request: An **Origin** field is added to the header information.  
  - Not-so-simple request: An HTTP request is sent before formal communication.  
  If CORS (not-so-simple request) is enabled for an API, another API that uses the OPTIONS method must be created. For details, see **Enabling 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 4-4**.
Figure 4-6 Adding input parameters

Table 4-4 Input parameter definition

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Name        | Name of the input parameter. If the parameter location is set to PATH, the parameter name must be 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 Location is specified as 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 include **PATH**, **HEADER**, and **QUERY**. |
| Type        | Type of the parameter value. Options: **STRING** and **NUMBER**.             |
| Mandatory   | Configure 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 default 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 automatically sends the default value to the backend service. |
| Example     | Example value of the parameter.                                            |
| Description | Description of the parameter.                                              |
3. Click OK.

**Step 3** If you set Method to POST, PUT, PATCH, or ANY, set the request body.

**Step 4** Click Next.

----End

**Defining Backend Request**

**Step 1** On the Define Backend Request page, select a backend type.

API Gateway supports HTTP/HTTPS, FunctionGraph, and Mock. Table 4-5 lists the parameters required for defining each type of backend service.

![HTTP/HTTPS backend figure]

**Table 4-5 Backend request parameters**

<table>
<thead>
<tr>
<th>Backend Type</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP/HTTPS</td>
<td>Protocol</td>
<td>Select HTTP or HTTPS. HTTPS is recommended for transmitting important or sensitive data. <strong>NOTE</strong> WebSocket is supported for HTTP and HTTPS.</td>
</tr>
<tr>
<td></td>
<td>Method</td>
<td>API calling method. The options include GET, POST, DELETE, PUT, PATCH, HEAD, OPTIONS, and ANY. ANY indicates that the API can be accessed using any request method.</td>
</tr>
<tr>
<td></td>
<td>Configure VPC Channel</td>
<td>Indicates whether to access the backend service using a VPC channel. You can configure the API to use or not to use a VPC channel.</td>
</tr>
<tr>
<td>Backend Type</td>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Backend Address (if applicable)</td>
<td>Set this parameter if no VPC channel is used. Enter a backend address in the format of &quot;host IP address or domain name&quot;:&quot;port number&quot;. The default port (80 for HTTP and 443 for HTTPS) will be used if no port is specified. The port number ranges from 1 to 65535. To use a variable, enclose the variable name in number signs (#), for example, #ipaddress#. Multiple variables can be used, for example, #ipaddress##test#.</td>
</tr>
<tr>
<td></td>
<td>VPC Channel (if applicable)</td>
<td>Select a VPC channel. Set this parameter if a VPC channel is used.</td>
</tr>
<tr>
<td></td>
<td>Host Header (if applicable)</td>
<td>Define a host header for requests to be sent to Elastic Cloud Servers (ECSs) associated with the VPC channel. By default, the original host header in each request is used. Set this parameter if a VPC channel is used.</td>
</tr>
<tr>
<td></td>
<td>Path</td>
<td>Path (URI) of the backend service. A path can contain parameters, which must be enclosed in braces ({ }). For example, /getUserInfo/{userId}. If the path contains an environment variable, enclose the environment variable in number signs (#), for example, / #path#. Multiple environment variables can be used, for example, /#path##request#.</td>
</tr>
<tr>
<td></td>
<td>Timeout (ms)</td>
<td>Request timeout duration. Range: 1 ~ 60,000 ms. The default value is 5000 ms.</td>
</tr>
<tr>
<td></td>
<td>FunctionGraph</td>
<td>Function URN ID of the requested function. Click Select Function URN to specify a function URN.</td>
</tr>
<tr>
<td></td>
<td>Version</td>
<td>Version of the function.</td>
</tr>
<tr>
<td></td>
<td>Invocation Mode</td>
<td><strong>Synchronous</strong>: synchronous invocation. When receiving an invocation request, FunctionGraph processes the request immediately and returns a result. The client closes the connection once receiving a response from the backend. <strong>Asynchronous</strong>: asynchronous invocation. Clients do not care about the function invocation results of their requests. Once receiving a request, FunctionGraph puts it in a queue, returns a response, and then processes requests one by one in idle state.</td>
</tr>
<tr>
<td></td>
<td>Timeout (ms)</td>
<td>Request timeout duration. Range: 1 ~ 60,000 ms. The default value is 5000 ms.</td>
</tr>
</tbody>
</table>
Mock

<table>
<thead>
<tr>
<th>Backend Type</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mock</td>
<td>Response</td>
<td>Mock is used for API development, debugging, and verification. It enables API Gateway to return a response without sending the request to the backend. This is useful for testing APIs when the backend is not available.</td>
</tr>
</tbody>
</table>

**NOTE**
- APIs that use variables cannot be debugged.
- Variable names are case-sensitive.
- For variables defined during API creation, corresponding variables and their values must be created. Otherwise, the APIs cannot be published because there will be no values that can be assigned to the variables.

**Step 2** (Optional) Add a backend policy.

Backend policies can be added to forward requests to different backend services.

1. Click **Add Backend Policy**.
2. Set parameters by referring to **Table 4-6** and **Table 4-5**.

**Figure 4-8** Setting backend policy parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Backend policy name.</td>
</tr>
</tbody>
</table>
| 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 to make the backend policy take effect. Set conditions by referring to **Table 4-7**. |
### Table 4-7 Policy conditions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Source    | - Source IP address  
    - Input parameter |

Parameter Name

This parameter is required only when **Source** is set to **Input parameter**. Select an input parameter.

Condition Type

This parameter is required only when **Source** is set 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 in the regularly 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 **Source** is set to **Source IP address**, enter one or more IP addresses and separate them using commas.

---

**Step 3**  (Optional) Set backend parameters.

Map the input parameters to backend parameters in backend requests.

1. Click next to **Backend Parameters**, and define backend parameters in either of the following two ways:
   - 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 4-9** highlights the backend parameters.
Figure 4-9 Configuring backend parameters

a. If the parameter location is set to **PATH**, the parameter name must be 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 **Location** is specified as **HEADER**, ensure that the parameter name does not contain underscores (_).

c. As shown in Figure 4-9, parameters **test01** and **test03** are respectively 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:**

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

**Backend request:**

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

**Step 4** (Optional) Set constant parameters.

Constant parameters can be defined to receive constants that are invisible to API callers. When requesting a backend service, API Gateway adds constant parameters to specified positions in the request and then sends the request to the backend service.

1. Click **next to Constant Parameters**.
2. Click **Add Constant Parameter**, and set the parameters listed in **Table 4-8**.

Figure 4-10 Adding constant parameters
Table 4-8 Setting constant parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Name      | Name of the constant parameter. If the parameter location is set to **PATH**, the parameter name must be 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 **Location** is specified as **HEADER**, ensure that the parameter name does not contain underscores `_`.

<table>
<thead>
<tr>
<th>Location</th>
<th>Position of the parameter in requests. The options include <strong>PATH</strong>, <strong>QUERY</strong>, and <strong>HEADER</strong>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Value of the parameter.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the constant parameter.</td>
</tr>
</tbody>
</table>

**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 parameters, the following characters will be percent-encoded: ASCII codes 0 - 31, blank symbols, ASCII codes 127 - 255, and special characters `?<>%#"[]^`.

**Step 5** Click **Next**.

---End

### Defining Responses

**Step 1** On the **Define Response** page, set the parameters listed in **Table 4-9**.

Table 4-9 Defining responses

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example Success Response</td>
<td>Example of a response returned when the API is called successfully</td>
</tr>
<tr>
<td>Example Failure Response</td>
<td>Example of a response returned when the API fails to be called</td>
</tr>
</tbody>
</table>

**Step 2** Click **Finish**.
After the API is created, click its name in the API list to view details.

---End

Creating an API Using a REST API

Alternatively, create your own API by using a REST API provided by API Gateway. For more information, see the following reference.

Registering an API

FAQs About API Creation

- Does API Gateway Support Multiple Backend Endpoints?
- How Do I Choose an Authentication Mode?
- What Are the Possible Causes If a Backend Service Fails to Be Invoked?
- What Can I Do If "No backend available." Is Displayed When I Call an API?
- What Can I Do If "The API does not exist or has not been published in the environment." Is Displayed When I Call an API with JavaScript?

Best Practice for Creating an API

- Querying Mobile Number Home Locations

Follow-Up Operations

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

4.5 Debugging an API

Scenarios

After creating an API, debug it on the API Gateway console by adding HTTP headers and body parameters to verify whether the API service is normal.

**NOTE**

- APIs with backend request paths containing variables cannot be debugged.
- APIs with bound signature keys cannot be debugged.
- Request throttling policies bound to an API do not work during debugging.

Prerequisites

- An API group and API have been created.
- The backend service of the API has been set up.

Procedure

**Step 1** Log in to the management console.
**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > APIs.

**Step 5** Debug an API in either of the following two ways:
- 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.

![Figure 4-11 Debugging an API](image)

On the left side, set API request parameters listed in **Table 4-10**. On the right side, view the API request and response information after you click Send Request.

**Table 4-10 Parameters for debugging an API**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>This parameter can be modified only when Protocol has been set to HTTP&amp;HTTPS for the API.</td>
</tr>
<tr>
<td>Method</td>
<td>This parameter can be modified only when Method has been set to ANY for the API.</td>
</tr>
<tr>
<td>Suffix</td>
<td>You can define a path only when Matching has been set to Prefix match for the API.</td>
</tr>
<tr>
<td>Path Parameters</td>
<td>This parameter can be modified only when the value of Path contains braces ({}).</td>
</tr>
<tr>
<td>Header Parameters</td>
<td>HTTP header parameters and values.</td>
</tr>
<tr>
<td>Query Parameters</td>
<td>Query parameters and values.</td>
</tr>
<tr>
<td>Body</td>
<td>This parameter can be modified only when Method has been set to ANY, PATCH, POST, or PUT for the API.</td>
</tr>
</tbody>
</table>
Step 6 After setting request parameters, click **Send Request**.

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

- If the debugging is successful, the HTTP status code **200** and response information are displayed.
- If the debugging failed, the HTTP status code **4xx** or **5xx** is displayed. For details, see **Error Codes**.

Step 7 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.

Follow-Up Operations

After the API is successfully debugged, **publish** it in an environment so that it can be called by users. To ensure security of your API services, 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 APIs.

### 4.6 (Optional) Creating an Environment and Environment Variable

#### Scenarios

An API can be called in different environments, such as the production, test, and development environments. RELEASE is the default environment provided by API Gateway. Environment variables can be defined 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 defined during API creation, corresponding variables and their values must be created. For example, variable **Path** is defined for an API, and two variables with the same name are respectively created and assigned values **/Stage/test** and **/Stage/AA** in environments 1 and 2. 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 4-12 Application of environment variables**
NOTE

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

Prerequisites

An API group has been created.

Creating an Environment

Step 1  Log in to the management console.

Step 2  Click in the upper left corner to select a region.

Step 3  Choose Service List > Application > API Gateway.

Step 4  In the navigation pane, choose API Publishing > Environments.

Step 5  Click Create Environment, and set the parameters listed in Table 4-11.

![Creating an environment](image)

Table 4-11 Environment information

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Environment name</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the environment</td>
</tr>
</tbody>
</table>

Step 6  Click OK.

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

----End

Accessing an Environment

The default RELEASE environment can be accessed using a RESTful API. To access other environments, add the X-Stage header to the request to specify an environment name. For example, to access the DEVELOP environment, add X-Stage: DEVELOP to the request header.
Creating an Environment Variable

**Step 1** Log in to the management console.

**Step 2** Click the location in the upper left corner to select a region.

**Step 3** Choose **Service List > Application > API Gateway**.

**Step 4** In the navigation pane, choose **API Publishing > API Groups**.

**Step 5** Create a variable in either of the following two ways:

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

**Step 7** Set the parameters listed in **Table 4-12**.

**Figure 4-14** Creating an environment variable

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the variable you want to create. The name must be the same as that of the variable defined for the API.</td>
</tr>
<tr>
<td>Value</td>
<td>The path to be used in the selected environment.</td>
</tr>
</tbody>
</table>

**Step 8** Click **OK**.

**NOTE**

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

--- End

Creating an Environment and Variable Using REST APIs

Alternatively, create an environment and variable by using REST APIs provided by API Gateway. For more information, see the following references.
Creating an Environment

Creating an Environment Variable

FAQs About Environment and Variable Creation

Can I Invoke Different Backend Services by Publishing an API in Different Environments?

Follow-Up Operations

After creating an environment and variable, publish APIs in the environment for API callers to invoke.

4.7 Publishing an API

Scenarios

APIs can be called only after being published in an environment. An API can be published in different environments through its entire lifecycle. API Gateway provides the publication history (such as version, description, time, and environment) of each API, and supports rollback to different historical versions.

**NOTE**

- If a published API is modified, it must be published 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 can be retained for each environment.

Prerequisites

- An API group and API have been created.
- An environment has been created.

Publishing an API

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > APIs.

**Step 5** Publish an API in either of the following two ways:

- Click Publish in the same row as 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. A maximum of 1000 APIs can be published at a time.
Step 6  Select the environment in which the API will be published, and enter a description.

**Figure 4-15 Publishing an API**

![API Publishing interface](image)

**NOTE**
- If the API has already been published in the environment, publishing it again will overwrite its definition in the environment.
- You can also create a new environment if there is no environment that can meet your requirements.

Step 7  Click **Publish**.

-----End

**Viewing Publication History**

Step 1  Log in to the management console.

Step 2  Click ![location icon] in the upper left corner to select a region.

Step 3  Choose **Service List > Application > API Gateway**.

Step 4  In the navigation pane, choose **API Publishing > APIs**.

Step 5  Click the name of the target API.

Step 6  Click the **Publication History** tab.

The publication history of the API is displayed.

**Figure 4-16 Viewing publication history**

![Publication History](image)
Step 7  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 8**  To roll the API back to a historical version, click **Switch Version** in the same row as the target version, and click **Yes**.

Click **OK**. If "current version" is displayed next to the target version, the rollback is successful.

When the API is called, configuration of the current version rather than those last saved for the API is used.

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

**Publishing an API Using a REST API**

Alternatively, publish an API by using a REST API provided by API Gateway. For more information, see the following reference.

*Publishing an API*

**FAQs About API Publishing**

*Do I Need to Publish an API Again After Modification?*

*What Can I Do If an API Published in a Non-RELEASE Environment Is Inaccessible?*

*Can I Invoke Different Backend Services by Publishing an API in Different Environments?*
5 API Group Management

5.1 Creating an API Group

Scenarios

Before creating an API, you must create an API group. An API group facilitates management of APIs used for the same service.

**NOTE**

- Each API belongs to only 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 to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > API Groups.

**Step 5** Click Create API Group, and set the parameters listed in Table 5-1 according to Figure 5-1.
Figure 5-1 Creating an API group

Create API Group

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>API group name</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the API group</td>
</tr>
</tbody>
</table>

Table 5-1 Parameters for creating an API group

Step 6  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 open APIs, you must bind independent domain names to the API group to make the APIs accessible to users.

---End

Creating an API Group Using a REST API

Alternatively, create an API group by using a REST API provided by API Gateway. For more information, see the following reference.

Creating an API Group

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 invoke APIs in the group. For more information, see Binding a Domain Name.
5.2 Binding a Domain Name

Scenarios

Before an API is opened, one or more independent domain names must be bound 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 each API group.

Before binding a domain name, understand the following two concepts:

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

Prerequisites

1. The independent domain name to be bound has been CNAMEd to the subdomain name of the API group. For details, see [Adding a CNAME Record Set](#).
2. If the API group contains APIs that are called through HTTPS, SSL certificates need to be added to 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 to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > API Groups.

**Step 5** Bind a domain name in either of the following two ways:

- 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 6** Click Bind Domain Name, enter a domain name, and click ✓. Figure 5-2 shows the successfully bound domain name.
Figure 5-2 Successfully bound domain name

NOTE
If the domain name is not needed, click Unbind in the same row to unbind it from the API group.

Step 7  (Optional) If the API group contains APIs that are called through HTTPS, add SSL certificates to the group.
1. Click Add SSL Certificate.
2. Enter a certificate name, certificate content, and a key, and click OK.

NOTE
- To edit the 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 the SSL certificate is not needed, click Delete SSL Certificate in the same row to delete it.

Binding a Domain Name Using a REST API

Alternatively, bind an independent domain name to an API group by using a REST API provided by API Gateway. For more information, see the following references.

Binding a Domain Name

Binding a Certificate

Follow-Up Operations

After binding independent domain names to an API group, create APIs in the group to open up your backend capabilities. For details, see Creating an API.

5.3 Deleting an API Group

Scenarios

Delete an API group if it is not needed.
NOTE
API groups that have been listed on the marketplace cannot be deleted.

Prerequisites
An API group has been created.

Procedure

Step 1  Log in to the management console.

Step 2  Click in the upper left corner to select a region.

Step 3  Choose Service List > Application > API Gateway.

Step 4  In the navigation pane, choose API Publishing > API Groups.

Step 5  Delete an API group in either of the following two ways:
   ○ In the Operation column of the target API group, choose More > Delete.
   ○ Click the name of the target API group, and click Delete in the upper right corner of the displayed API group details page.

Step 6  Enter DELETE and click Yes.

---End

Deleting an API Group Using a REST API

Alternatively, delete an API group by using a REST API provided by API Gateway. For more information, see the following reference.

Deleting an API group
6 API Management

6.1 Creating an API

Scenarios

Create an API in API Gateway to open up your service capabilities.

To create an API, you need to set the basic information, and define API request, backend request, and responses.

**NOTE**

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

Prerequisites

- An API group has been created. If no API groups are available, click Create API Group to create one on the Set Basic Information page.
- If the backend service is deployed in a VPC, create a VPC channel for accessing the service by following the procedure in Creating a VPC Channel or by clicking Manage VPC Channel on the Define Backend Request page.

Setting Basic Information

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > APIs.

**Step 5** Click Create API, and set the parameters listed in Table 6-1 according to Figure 6-1.
Figure 6-1 Setting the basic information

![Setting the basic information](image)

Table 6-1 Basic information

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>API name</td>
</tr>
<tr>
<td>API Group</td>
<td>Group to which the API belongs</td>
</tr>
<tr>
<td></td>
<td>If no API groups are available, click <strong>Create API Group</strong> to create one.</td>
</tr>
<tr>
<td>Visibility</td>
<td>Configure whether the API is available to the public. Options:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Public</strong>: The API can be released to the marketplace.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Private</strong>: The API is excluded when the API group to which it belongs is released to the marketplace.</td>
</tr>
<tr>
<td>Security Authentication</td>
<td>There are three authentication modes:</td>
</tr>
<tr>
<td></td>
<td>• <strong>App</strong>: API Gateway authenticates API requests.</td>
</tr>
<tr>
<td></td>
<td>• <strong>IAM</strong>: Identity and Access Management (IAM) authenticates API requests.</td>
</tr>
<tr>
<td></td>
<td>• <strong>None</strong>: No authentication is required.</td>
</tr>
<tr>
<td></td>
<td>The API calling method varies depending on the authentication mode. For details, see the <strong>Developer Guide</strong>.</td>
</tr>
<tr>
<td></td>
<td>The App authentication mode is recommended.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTICE</strong></td>
</tr>
<tr>
<td></td>
<td>• If the authentication mode of the API is set to <strong>IAM</strong>, any HUAWEI CLOUD tenant can access the API, which may result in excessive charges caused by malicious requests.</td>
</tr>
<tr>
<td></td>
<td>• If the authentication mode of the API is set to <strong>None</strong>, any public network user can access the API, which may also result in excessive charges caused by malicious requests.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the API</td>
</tr>
</tbody>
</table>
**Step 6**  Click Next.

---End

**Defining API Request**

**Step 1**  On the **Define API Request** page, set parameters according to **Table 6-2**.

![Figure 6-2 Defining API request](image)

**Table 6-2 API request parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain Name</td>
<td>Subdomain automatically allocated by the system to the API group.</td>
</tr>
</tbody>
</table>
| Protocol    | Protocol used for invoking the API. Options:
  - HTTP
  - HTTPS
  - HTTP&HTTPS
  HTTPS is recommended for transmitting important or sensitive data. |
| Path        | Path for requesting the API
  Enter a path in the format of /{serviceName}/{interfaceName}.
  - The content in braces ({}) is case-sensitive.
  - Parameters contained in the request path must be defined as input parameters. |
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Matching** | Options:  
- **Exact match**: The API can be invoked using the specified request path.  
- **Prefix match**: The API can be invoked using paths starting with the matching characters.  
For example, if the request path is set to `/test/AA` and the matching mode to **Prefix match**, the API can be invoked using `/test/AA/CC` but cannot be invoked using `/test/AACC`.  
**NOTE**  
The plus (+) sign is not supported for prefix match. |
| **Method** | API calling method. The options include **GET**, **POST**, **DELETE**, **PUT**, **PATCH**, **HEAD**, **OPTIONS**, and **ANY**.  
**ANY** indicates that the API can be accessed using any request method. |
| **CORS** | Enables or disables 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.  
CORS requests are classified into two types:  
- Simple request: An **Origin** field is added to the header information.  
- Not-so-simple request: An HTTP request is sent before formal communication.  
If CORS (not-so-simple request) is enabled for an API, another API that uses the OPTIONS method must be created. For details, see **Enabling 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 6-3**.
### Table 6-3 Input parameter definition

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Name        | Name of the input parameter. If the parameter location is set to **PATH**, the parameter name must be 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 **Location** is specified as **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 include **PATH**, **HEADER**, and **QUERY**. |
| Type        | Type of the parameter value. Options: **STRING** and **NUMBER**. |
| Mandatory   | Configure 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 default 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 automatically sends the default value to the backend service. |
| Example     | Example value of the parameter. |
| Description | Description of the parameter. |
3. Click OK.

**Step 3** If you set Method to POST, PUT, PATCH, or ANY, set the request body.

**Step 4** Click Next.

---End

### Defining Backend Request

**Step 1** On the Define Backend Request page, select a backend type.

API Gateway supports HTTP/HTTPS, FunctionGraph, and Mock. Table 6-4 lists the parameters required for defining each type of backend service.

**Figure 6-4 HTTP/HTTPS backend**

![HTTP/HTTPS backend](image)

**Table 6-4 Backend request parameters**

<table>
<thead>
<tr>
<th>Backend Type</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP/HTTPS</td>
<td>Protocol</td>
<td>Select HTTP or HTTPS. HTTPS is recommended for transmitting important or sensitive data. <strong>NOTE</strong> WebSocket is supported for HTTP and HTTPS.</td>
</tr>
<tr>
<td></td>
<td>Method</td>
<td>API calling method. The options include GET, POST, DELETE, PUT, PATCH, HEAD, OPTIONS, and ANY. ANY indicates that the API can be accessed using any request method.</td>
</tr>
<tr>
<td>Configure VPC Channel</td>
<td></td>
<td>Indicates whether to access the backend service using a VPC channel. You can configure the API to use or not to use a VPC channel.</td>
</tr>
<tr>
<td>Backend Type</td>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>Backend Address (if applicable)</td>
<td>Set this parameter if no VPC channel is used. Enter a backend address in the format of &quot;host IP address or domain name&quot;:&quot;port number&quot;. The default port (80 for HTTP and 443 for HTTPS) will be used if no port is specified. The port number ranges from 1 to 65535. To use a variable, enclose the variable name in number signs (#), for example, #ipaddress#. Multiple variables can be used, for example, #ipaddress##test#.</td>
<td></td>
</tr>
<tr>
<td>VPC Channel (if applicable)</td>
<td>Select a VPC channel. Set this parameter if a VPC channel is used.</td>
<td></td>
</tr>
<tr>
<td>Host Header (if applicable)</td>
<td>Define a host header for requests to be sent to Elastic Cloud Servers (ECSs) associated with the VPC channel. By default, the original host header in each request is used. Set this parameter if a VPC channel is used.</td>
<td></td>
</tr>
<tr>
<td>Path</td>
<td>Path (URI) of the backend service. A path can contain parameters, which must be enclosed in braces ({}). For example, /getUserInfo/{userId}. If the path contains an environment variable, enclose the environment variable in number signs (#), for example, / #path#. Multiple environment variables can be used, for example, /#path##request#.</td>
<td></td>
</tr>
<tr>
<td>Timeout (ms)</td>
<td>Request timeout duration. Range: 1 – 60,000 ms. The default value is 5000 ms.</td>
<td></td>
</tr>
<tr>
<td>FunctionGraph</td>
<td>ID of the requested function. Click Select Function URN to specify a function URN.</td>
<td></td>
</tr>
<tr>
<td>Version</td>
<td>Version of the function.</td>
<td></td>
</tr>
<tr>
<td>Invocation Mode</td>
<td><strong>Synchronous</strong>: synchronous invocation. When receiving an invocation request, FunctionGraph processes the request immediately and returns a result. The client closes the connection once receiving a response from the backend. <strong>Asynchronous</strong>: asynchronous invocation. Clients do not care about the function invocation results of their requests. Once receiving a request, FunctionGraph puts it in a queue, returns a response, and then processes requests one by one in idle state.</td>
<td></td>
</tr>
<tr>
<td>Timeout (ms)</td>
<td>Request timeout duration. Range: 1 – 60,000 ms. The default value is 5000 ms.</td>
<td></td>
</tr>
<tr>
<td>Backend Type</td>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Mock</td>
<td>Response</td>
<td>Mock is used for API development, debugging, and verification. It enables API Gateway to return a response without sending the request to the backend. This is useful for testing APIs when the backend is not available.</td>
</tr>
</tbody>
</table>

**Table 6-5 Backend policy parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Backend policy name.</td>
</tr>
</tbody>
</table>
| 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 to make the backend policy take effect. Set conditions by referring to Table 6-6. |
### Table 6-6 Policy conditions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Source      | - Source IP address  
|             | - Input parameter                                 |
| Parameter Name | This parameter is required only when Source is set to Input parameter. Select an input parameter. |
| Condition Type | This parameter is required only when Source is set 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 in the regularly 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 Source is set to Source IP address, enter one or more IP addresses and separate them using commas. |

**Step 3** (Optional) Set backend parameters.

Map the input parameters to backend parameters in backend requests.

1. Click next to **Backend Parameters**, and define backend parameters in either of the following two ways:
   - 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 6-6** highlights the backend parameters.
**Figure 6-6 Configuring backend parameters**

- If the parameter location is set to **PATH**, the parameter name must be the same as that defined in the backend request path.
- 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 Location is specified as `HEADER`, ensure that the parameter name does not contain underscores `_`.
- As shown in **Figure 6-6**, parameters test01 and test03 are respectively 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:**
```bash
curl -ik -H 'test02: def' -X GET https://myhwclouds.com/v1.0/abc?test03=xyz
```

**Backend request:**
```bash
curl -ik -H 'test01: abc' -H 'test03: xyz' -X GET https://myhwclouds.com/v1.0/def
```

**Step 4** (Optional) Set constant parameters.

Constant parameters can be defined to receive constants that are invisible to API callers. When requesting a backend service, API Gateway adds constant parameters to specified positions in the request and then sends the request to the backend service.

1. Click ▼ next to **Constant Parameters**.
2. Click **Add Constant Parameter**, and set the parameters listed in Table 6-7.

**Figure 6-7 Adding constant parameters**
**Table 6-7 Setting constant parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the constant parameter. If the parameter location is set to <strong>PATH</strong>, the parameter name must be the same as that defined in the backend request path. <strong>NOTE</strong>&lt;br&gt;- The parameter name is not case-sensitive. It cannot start with <code>x-apig-</code> or <code>x-sdk-</code>. &lt;br&gt;- The parameter name cannot be <code>x-stage</code>. &lt;br&gt;- If <strong>Location</strong> is specified as <strong>HEADER</strong>, ensure that the parameter name does not contain underscores <code>_</code>.</td>
</tr>
<tr>
<td>Location</td>
<td>Position of the parameter in requests. The options include <strong>PATH</strong>, <strong>QUERY</strong>, and <strong>HEADER</strong>.</td>
</tr>
<tr>
<td>Value</td>
<td>Value of the parameter.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the constant parameter.</td>
</tr>
</tbody>
</table>

**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 parameters, the following characters will be percent-encoded: ASCII codes 0 – 31, blank symbols, ASCII codes 127 – 255, and special characters `>=<+&%#^[^|]{}`

**Step 5** Click Next.

End

**Defining Responses**

**Step 1** On the **Define Response** page, set the parameters listed in **Table 6-8**.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example Success Response</td>
<td>Example of a response returned when the API is called successfully</td>
</tr>
<tr>
<td>Example Failure Response</td>
<td>Example of a response returned when the API fails to be called</td>
</tr>
</tbody>
</table>

**Step 2** Click **Finish**.
After the API is created, click its name in the API list to view details.

---End

**Creating an API Using a REST API**

Alternatively, create your own API by using a REST API provided by API Gateway. For more information, see the following reference.

**Registering an API**

**FAQs About API Creation**

- Does API Gateway Support Multiple Backend Endpoints?
- How Do I Choose an Authentication Mode?
- What Are the Possible Causes If a Backend Service Fails to Be Invoked?
- What Can I Do If "No backend available." Is Displayed When I Call an API?
- What Can I Do If "The API does not exist or has not been published in the environment." Is Displayed When I Call an API with JavaScript?

**Best Practice for Creating an API**

**Querying Mobile Number Home Locations**

**Follow-Up Operations**

After creating an API, verify the API service by following the procedure in [Debugging an API](#).

**6.2 Enabling CORS**

By default, CORS is disabled for each API created on API Gateway. If an API will be accessed from other domains rather than the API's own domain, you need to enable CORS for the API. For an API that will receive not-so-simple requests, you must create another API with the OPTIONS method to respond to preflight requests.

**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 for resource access.
Figure 6-8 Process flow of the CORS mechanism

CORS requests are classified into two types: simple requests and not-so-simple requests.

A simple request meets the following conditions:
- The request method is HEAD, GET, or POST.
- The request header contains only the following fields:
  - Accept
  - Accept-Language
  - Content-Language
  - Last-Event-ID
  - Content-Type (Only these three values are allowed: application/x-www-form-urlencoded, multipart/form-data, and 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 based on the origin whether the request is safe and can be accepted. If the server sends a response containing the Access-Control-Allow-Origin field, the server accepts the request.

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

Simple Request

When creating an API that will receive simple requests, enable CORS by following the instructions in Creating an API.

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

{"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 can handle 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
Origin: 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"}
```


**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
```

```
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 by following the instructions in Creating an API, and create a new 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**: Same as the group to which the API with CORS enabled belongs.
- **Security Authentication**: No authentication is required for requests received by the new API no matter which security authentication mode has been selected.
- **Protocol**: Same as the protocol used by the API with CORS enabled.
- **Path**: Same as or prefixally matching the path set for the API with CORS enabled.
- **Method**: Set to 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:

```plaintext
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 required to specify the additional header fields in the subsequent simple requests.

#### Response of 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: c9b89268888c356d6a9581c5c10bb4d11
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-Allow-Methods: GET,POST,PUT,DELETE,HEAD,OPTIONS,PATCH
Access-Control-Max-Age: 172800
```
6.3 Debugging an API

Scenarios

After creating an API, debug it on the API Gateway console by adding HTTP headers and body parameters to verify whether the API service is normal.

NOTE

- APIs with backend request paths containing variables cannot be debugged.
- APIs with bound signature keys cannot be debugged.
- Request throttling policies bound to an API do not work during debugging.
Prerequisites

- An API group and API have been created.
- The backend service of the API has been set up.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner to select a region.

Step 3 Choose Service List > Application > API Gateway.

Step 4 In the navigation pane, choose API Publishing > APIs.

Step 5 Debug an API in either of the following two ways:

- 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.

Figure 6-9 Debugging an API

On the left side, set API request parameters listed in Table 6-9. On the right side, view the API request and response information after you click Send Request.

Table 6-9 Parameters for debugging an API

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>This parameter can be modified only when Protocol has been set to HTTP&amp;HTTPS for the API.</td>
</tr>
<tr>
<td>Method</td>
<td>This parameter can be modified only when Method has been set to ANY for the API.</td>
</tr>
<tr>
<td>Suffix</td>
<td>You can define a path only when Matching has been set to Prefix match for the API.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path Parameters</td>
<td>This parameter can be modified only when the value of Path contains braces ({}).</td>
</tr>
<tr>
<td>Header Parameters</td>
<td>HTTP header parameters and values.</td>
</tr>
<tr>
<td>Query Parameters</td>
<td>Query parameters and values.</td>
</tr>
<tr>
<td>Body</td>
<td>This parameter can be modified only when Method has been set to ANY, PATCH, POST, or PUT for the API.</td>
</tr>
</tbody>
</table>

**Step 6**  
After setting request parameters, click **Send Request**.

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

- If the debugging is successful, the HTTP status code **200** and response information are displayed.
- If the debugging failed, the HTTP status code **4xx** or **5xx** is displayed. For details, see **Error Codes**.

**Step 7**  
Send more requests with different parameters and values to verify the API.

---

**Follow-Up Operations**

After the API is successfully debugged, **publish** it in an environment so that it can be called by users. To ensure security of your API services, 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 APIs.

### 6.4 Authorizing Apps to Call an API

**Scenarios**

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

---

**Prerequisites**

-  An API group and API have been created.
Procedure

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > APIs.

**Step 5** Authorize apps to call an API in any of the following ways:

- In the Operation column of the target API, click Authorize App, and then click Select App.
- Select one or more target APIs, 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. On the displayed API details page, click the Summary tab.
  c. Click the Authorization tab.
  d. Click Select App.

**NOTE**

To authorize an app to access multiple APIs, select the APIs, and click Authorize App. Click Select App, select an app, and click OK. Access to a maximum of 1000 APIs can be granted to an app at a time.

**Step 6** Select an environment, query and select desired apps, and click OK.

![Figure 6-10 Authorizing an app to call an API](image)

**Step 7** After the authorization is complete, view the authorized apps on the Authorization tab page or the Authorize App page.

**NOTE**

If an app is not needed to call the API, click Cancel Authorization in the same row to unbind the app.

---End

**Authorizing an App to Call an API by Using a REST API**

Alternatively, authorize an app to call your own API by using a REST API provided by API Gateway. For more information, see the following references.
Authorizing Apps

Canceling Authorization

Follow-Up Operations

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

6.5 Publishing an API

Scenarios

APIs can be called only after being published in an environment. An API can be published in different environments through its entire lifecycle. API Gateway provides the publication history (such as version, description, time, and environment) of each API, and supports rollback to different historical versions.

NOTE

- If a published API is modified, it must be published 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 can be retained for each environment.

Prerequisites

- An API group and API have been created.
- An environment has been created.

Publishing an API

Step 1 Log in to the management console.

Step 2 Click in the upper left corner to select a region.

Step 3 Choose Service List > Application > API Gateway.

Step 4 In the navigation pane, choose API Publishing > APIs.

Step 5 Publish an API in either of the following two ways:

- Click Publish in the same row as 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. A maximum of 1000 APIs can be published at a time.

Step 6 Select the environment in which the API will be published, and enter a description.
**Figure 6-11 Publishing an API**

<table>
<thead>
<tr>
<th>API Name</th>
<th>QueryPhoneNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environment</td>
<td>RELEASE</td>
</tr>
</tbody>
</table>

**NOTE**
- If the API has already been published in the environment, publishing it again will overwrite its definition in the environment.
- You can also create a new environment if there is no environment that can meet your requirements.

**Step 7** Click **Publish**.

----End

**Viewing Publication History**

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose **Service List > Application > API Gateway**.

**Step 4** In the navigation pane, choose **API Publishing > APIs**.

**Step 5** Click the name of the target API.

**Step 6** Click the **Publication History** tab.

The publication history of the API is displayed.

**Figure 6-12 Viewing publication history**

**Step 7** 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 8** To roll the API back to a historical version, click **Switch Version** in the same row as the target version, and click **Yes**.

Click **OK**. If "current version" is displayed next to the target version, the rollback is successful.

When the API is called, configuration of the current version rather than those last saved for the API is used.

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.

---

**Publishing an API Using a REST API**

Alternatively, publish an API by using a REST API provided by API Gateway. For more information, see the following reference.

**Publishing an API**

**FAQs About API Publishing**

**Do I Need to Publish an API Again After Modification?**

**What Can I Do If an API Published in a Non-RELEASE Environment Is Inaccessible?**

**Can I Invoke Different Backend Services by Publishing an API in Different Environments?**

### 6.6 Taking an API Offline

**Scenarios**

If a published API is not needed to provide services, you can remove it from the environment in which it has been published.

---

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

---

**Prerequisites**

- An API group and API have been created.
- The API has been published in an environment.
Procedure

Step 1  Log in to the management console.

Step 2  Click in the upper left corner to select a region.

Step 3  Choose Service List > Application > API Gateway.

Step 4  In the navigation pane, choose API Publishing > APIs.

Step 5  Take an API offline in either of the following two ways:
   - In the Operation column of the target API, choose More > Take Offline.
   - Click the name of the target API, click the name of the target API, and click Take Offline in the upper right corner of the displayed API details page.

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

Step 6  Select the environment from which you want to take the API offline, and click Yes.

---End

Taking an API Offline Using a REST API

Alternatively, take your own API offline by using a REST API provided by API Gateway. For more information, see the following reference.

Taking an API Offline

Follow-Up Operations

After taking an API offline, delete the API according to Deleting an API to release the resources used by the API.

6.7 Deleting an API

Scenarios

You can delete a published API if it is not needed to provide services.

**NOTICE**

- Deleted APIs cannot be accessed by apps and users that were using the APIs. Ensure you have notified users before the deletion.
- APIs that have already been listed on the marketplace cannot be deleted.
- Published APIs must be taken offline before being deleted.
Prerequisites

- An API group and API have been created.
- The API is not needed to provide services.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner to select a region.

Step 3 Choose Service List > Application > API Gateway.

Step 4 In the navigation pane, choose API Publishing > APIs.

Step 5 Delete an API in either of the following two ways:

- 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. A maximum of 1000 APIs can be deleted at a time.

Step 6 Enter DELETE and click Yes.

----End

Deleting an API Using a REST API

Alternatively, delete an API by using a REST API provided by API Gateway. For more information, see the following reference.

Deleting an API

6.8 Importing APIs

Scenarios

Swagger is an open-source software framework compliant with OpenAPI specifications. It helps developers design, build, document, and use REST APIs. API Gateway supports the import of APIs defined in Swagger 2.0 files into an existing or a new API group.

APIs can be imported one by one 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 Extended Definition.
- Your API group and API quotas are sufficient.
Procedure

**Step 1**  Log in to the management console.

**Step 2**  Click in the upper left corner to select a region.

**Step 3**  Choose Service List > Application > API Gateway.

**Step 4**  In the navigation pane, choose API Publishing > APIs.

**Step 5**  Click Import API.

**Step 6**  Set the parameters listed in Table 6-10.

![Figure 6-13 Importing APIs](image)

**Table 6-10 Parameters for importing APIs**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import To</td>
<td>Options:</td>
</tr>
<tr>
<td></td>
<td>- <strong>New API group</strong>: Import APIs to a new API group. If you select this option, the system automatically creates a new API group and imports the APIs to this group.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Existing API group</strong>: Import APIs to an existing API group. If you select this option, the system adds the APIs to the selected group while retaining the existing APIs in it.</td>
</tr>
<tr>
<td>API Group</td>
<td>Select an API group if you set Import To to <strong>Existing API group</strong>.</td>
</tr>
<tr>
<td>Overwrite</td>
<td>Set this parameter if you set Import To to <strong>Existing API group</strong>. After you enable this function, if the definition of an API you are importing is the same as that of an existing API, the new definition will overwrite the existing one.</td>
</tr>
<tr>
<td>Swagger</td>
<td>Select the Swagger file you want to import.</td>
</tr>
</tbody>
</table>

**Step 7**  Click Import, and view the imported APIs in the API group.

**NOTE**

Imported APIs must be manually published before they are available for users to access.
Follow-Up Operations

Publish the imported API in an environment for users to invoke.

6.9 Exporting APIs

Scenarios

Export APIs as JSON or YAML files. APIs can be exported one by one or in batches as required.

NOTE

APIs purchased in the marketplace cannot be exported.

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 to select a region.

Step 3  Choose Service List > Application > API Gateway.

Step 4  Click Export API.

Step 5  Set the parameters listed in Table 6-11.

Figure 6-14 Exporting APIs

Table 6-11 Parameters for exporting APIs

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API Group</td>
<td>Select the API group from which APIs are to be exported.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Environment</td>
<td>Select the environment in which the APIs to be exported have been published.</td>
</tr>
<tr>
<td>APIs</td>
<td>By default, all APIs of the API group that have been published in the selected environment are exported. To export specified APIs, click Select API, and specify the APIs you want to export.</td>
</tr>
</tbody>
</table>
| API Definition| - **Basic**: The basic definition of an API is composed of the request and response definitions and 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 as **JSON** or **YAML** format.                                                                                                                                                                     |
| Version       | Set the version of the APIs to be exported. If no version is specified, the version will be set to the current date and time.                                                                                     |

**Step 6**  Click **Export**. The export result is displayed on the right.

---End
7 Request Throttling

7.1 Creating a Request Throttling Policy

Scenarios

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

To provide continuous and stable services, create request throttling policies to control the number of calls made to your APIs.

Request throttling policies take effect for an API only after being bound to the API.

**NOTE**

- You can create a maximum of 30 request throttling policies.
- An API can be bound to only one request throttling policy in an environment, but each request throttling policy can be bound to multiple APIs.

Prerequisites

The API to which the request throttling policy will be bound has been published.

Creating a Request Throttling Policy

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > Request Throttling.

**Step 5** Click Create Request Throttling Policy, and set the parameters listed in Table 7-1.
**Figure 7-1** Creating a request throttling policy

![Image of request throttling policy form]

**Table 7-1** Parameters for creating a request throttling policy

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Request throttling policy name.</td>
</tr>
<tr>
<td>Period</td>
<td>Period of time for limiting the number of API calls. This parameter will apply with Max. API Requests, Max. User Requests, Max. App Requests, and Max. IP Address Requests.</td>
</tr>
<tr>
<td>Max. API Requests</td>
<td>Maximum number of times each of the bound APIs can be called within the specified period. This parameter will apply with Period.</td>
</tr>
<tr>
<td>Max. User Requests</td>
<td>Maximum number of times each of the bound APIs can be called by a user with the specified period. The value of this parameter must be less than that of Max. API Requests. This parameter will apply with Period.</td>
</tr>
<tr>
<td>Max. App Requests</td>
<td>Maximum number of times each of the bound APIs can be called by an app within the specified period. The value of this parameter must be less than that of Max. User Requests. This parameter will apply with Period.</td>
</tr>
<tr>
<td>Max. IP Address Requests</td>
<td>Maximum number of times each of the bound APIs can be called by an IP address within the specified period. The value of this parameter must be less than that of Max. API Requests. This parameter will apply with Period.</td>
</tr>
</tbody>
</table>
Parameter | Description
---|---
Description | Description of the request throttling policy.

**Step 6**  Click **OK**.

After the request throttling policy is created, it is displayed in the request throttling policy list. You can bind the request throttling policy to APIs to throttle API requests.

---End

**Binding a Request Throttling Policy to an API**

**Step 1**  Log in to the management console.

**Step 2**  Click [ ] in the upper left corner to select a region.

**Step 3**  Choose **Service List > Application > API Gateway**.

**Step 4**  In the navigation pane, choose **API Publishing > Request Throttling**.

**Step 5**  Bind a request throttling policy to an API in either of the following two ways:

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

**Step 6**  Specify an API group, environment, and API name keyword to filter the desired API.

**Step 7**  Select the API and click **OK**.

**Figure 7-2** Binding a request throttling policy to an API

![Select API](image)

**NOTE**

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

---End
Creating, Binding, and Unbinding a Request Throttling Policy Using APIs

Alternatively, create a request throttling policy, bind it to APIs, or unbind it from APIs by using REST APIs provided by API Gateway. For more information, see the following references.

Creating a Request Throttling Policy
Binding a Request Throttling Policy
Unbinding a Request Throttling Policy

Follow-Up Operations

To control the number of API calls received from a specific app, specify the app as an excluded one. For more information, see Adding an Excluded App or Tenant. After an excluded app is added, the threshold of the app overrides the app maximum settings of the request throttling policy, and the API and user maximum settings of the policy are still valid.

To control the number of API calls received from a specific tenant, specify the tenant as an excluded one. For more information, see Adding an Excluded App or Tenant. After an excluded tenant is added, the threshold of the tenant overrides the user maximum settings of the request throttling policy, and the API and app maximum settings of the policy are still valid.

7.2 Deleting a Request Throttling Policy

Scenarios

You can delete a request throttling policy if it is not needed to provide services.

Prerequisites

A request throttling policy has been created.

Procedure

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > Request Throttling.

**Step 5** Delete a request throttling policy in either of the following two ways:

- 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 the request throttling policy has been bound to APIs, unbind it from the APIs before deleting it.
- To delete multiple request throttling policies, select the policies, and click Delete. A maximum of 1000 request throttling policies can be deleted at a time.

Step 6 Click Yes.

Deleted Request Throttling Policy Using a REST API

Alternatively, delete a request throttling policy by using a REST API provided by API Gateway. For more information, see the following reference.

Deleting a Request Throttling Policy

7.3 Adding an Excluded App or Tenant

Scenarios

Add an excluded app or tenant to a request throttling policy to control the number of API calls received from a specific app or tenant.

Prerequisites

An app has been created.

Adding an Excluded App

Step 1 Log in to the management console.

Step 2 Click in the upper left corner to select a region.

Step 3 Choose Service List > Application > API Gateway.

Step 4 In the navigation pane, choose API Publishing > Request Throttling.

Step 5 Click the name of the target request throttling policy.

Step 6 On the displayed request throttling policy details page, click the Excluded Apps tab.

Step 7 Click Select Excluded App.

Step 8 Select an excluded app in either of the following two ways:

Figure 7-3 Adding an excluded app
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 not exceed the value of **Max. API Requests**.

---

### Adding an Excluded Tenant

**Step 1** Log in to the management console.

**Step 2** Click the username in the upper right corner and choose **My Credential** from the drop-down list.

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

*Figure 7-4* Viewing the account ID and project ID

**Step 4** Click in the upper left corner to select a region.

**Step 5** Choose **Service List > Application > API Gateway**.

**Step 6** In the navigation pane, choose **API Publishing > Request Throttling**.

**Step 7** Click the name of the target request throttling policy.

**Step 8** Click the **Excluded Tenants** tab.

**Step 9** Click **Select Excluded Tenant**.

**Step 10** In the **Select Excluded Tenant** dialog box, set the parameters listed in **Table 7-2**.

*Figure 7-5* Adding an excluded tenant

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tenant ID</td>
<td>Enter a tenant ID.</td>
</tr>
<tr>
<td>Threshold</td>
<td>Enter a threshold. per 1 minute</td>
</tr>
<tr>
<td>Max API Requests</td>
<td></td>
</tr>
</tbody>
</table>
Table 7-2 Excluded tenant configuration

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tenant ID</td>
<td>Account or project ID obtained in Step 3.</td>
</tr>
<tr>
<td></td>
<td>● App authentication: project ID</td>
</tr>
<tr>
<td></td>
<td>● IAM authentication: account ID</td>
</tr>
<tr>
<td>Threshold</td>
<td>Maximum number of times an API can be called by the tenant within a specified period.</td>
</tr>
<tr>
<td></td>
<td>The value of this parameter cannot exceed that of Max. API Requests.</td>
</tr>
</tbody>
</table>

Step 11  Click OK.

**NOTE**

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

For example, 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 (2 for tenant A and 4 for tenant B). If the request throttling policy is bound to an API, tenants A and B can access the API respectively 2 and 4 times within 1 minute.

---End

Adding an Excluded App or Tenant Using a REST API

Alternatively, add an excluded app or tenant to a request throttling policy by using a REST API provided by API Gateway. For more information, see the following reference.

Creating an Excluded Request Throttling Configuration

7.4 Removing an Excluded App or Tenant

Scenarios

Remove an excluded app or tenant of a request throttling policy if the excluded settings are not needed.

Prerequisites

- A request throttling policy has been created.
- An excluded app or tenant has been added 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 to select a region.

Step 3  Choose Service List > Application > API Gateway.

Step 4  In the navigation pane, choose API Publishing > Request Throttling.
Step 5  Click the name of the target request throttling policy.

Step 6  Click the **Excluded Apps** tab on the displayed request throttling policy details page.

Step 7  In the **Operation** column of the app you want to remove, click **Remove**.

Step 8  Click **Yes**.

---End

**Removing an Excluded Tenant**

Step 1  Log in to the management console.

Step 2  Click  in the upper left corner to select a region.

Step 3  Choose **Service List > Application > API Gateway**.

Step 4  In the navigation pane, choose **API Publishing > Request Throttling**.

Step 5  Click the name of the target request throttling policy.

Step 6  Click the **Excluded Tenants** tab.

Step 7  In the **Operation** column of the tenant you want to remove, click **Remove**.

Step 8  Click **Yes**.

---End

**Removing an Excluded App or Tenant Using a REST API**

Alternatively, remove an excluded app or tenant of a request throttling policy by using a REST API provided by API Gateway. For more information, see the following reference.

[Deleting an Excluded Request Throttling Configuration](#)
8 Access Control

8.1 Creating an Access Control Policy

Scenarios

Access control policies are one of the API security protection measures provided by API Gateway to allow or deny API access from specified IP addresses or accounts.

Access control policies take effect for an API only after being bound to the API.

NOTE
- You can create a maximum of 100 access control policies.
- An API can be bound with only one access control policy in the same 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  If you want to control API access from certain accounts, obtain the account names.

1. Click the username in the upper right corner and choose My Credential from the drop-down list.
2. View the account name on the My Credentials page.

Figure 8-1 Viewing the account name

My Credentials
Step 3  Click in the upper left corner to select a region.

Step 4  Choose Service List > Application > API Gateway.

Step 5  In the navigation pane, choose API Publishing > Access Control.

Step 6  Click Create Access Control Policy.

Step 7  In the Create Access Control Policy dialog box, set the parameters listed in Table 8-1.

Figure 8-2 Creating an access control policy

Table 8-1 Parameters for creating an access control policy

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Access control policy name</td>
</tr>
<tr>
<td>Restriction Type</td>
<td>Type of the source from which API calls are controlled</td>
</tr>
<tr>
<td>IP address</td>
<td>Specify IP addresses and IP address ranges that are allowed or not allowed to access an API.</td>
</tr>
<tr>
<td>Account name</td>
<td>Specify names of the accounts that are allowed or not allowed to access an API.</td>
</tr>
<tr>
<td>Effect</td>
<td>Options: Allow and Deny. This parameter is used along with Restriction Type to control the access of certain IP addresses or accounts to an API.</td>
</tr>
<tr>
<td>IP Address</td>
<td>IP addresses and IP address ranges that are allowed or not allowed to access an API. Set this parameter only when Restriction Type is set to IP address.</td>
</tr>
<tr>
<td>Account Names</td>
<td>Names of the accounts that are allowed or not allowed to access an API. You can enter multiple account names and separate them with commas, for example, aaa,bbb.</td>
</tr>
</tbody>
</table>
Step 8  Click OK.

---End

Binding an Access Control Policy to an API

Step 1  Log in to the management console.

Step 2  Click in the upper left corner to select a region.

Step 3  Choose Service List > Application > API Gateway.

Step 4  In the navigation pane, choose API Publishing > Access Control.

Step 5  Bind an access control policy to an API in either of the following two ways:

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

Step 6  Specify an API group, environment, and API name keyword to filter the desired API.

Step 7  Select the API and click OK.

**NOTE**

If an access control policy is no longer needed for an API, unbind it from the API. To unbind an access control policy from multiple APIs, select the APIs, and click Unbind. An access control policy can be unbound from a maximum of 1000 APIs at a time.

---End

8.2 Deleting an Access Control Policy

Scenarios

Delete an access control policy if it is not needed to provide services.

Prerequisites

An access control policy has been created.

Procedure

Step 1  Log in to the management console.

Step 2  Click in the upper left corner to select a region.

Step 3  Choose Service List > Application > API Gateway.

Step 4  In the navigation pane, choose API Publishing > Access Control.

Step 5  Delete an access control policy in either of the following two ways:

- 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 the access control policy has been bound to APIs, unbind it from the APIs before deleting it.
- To delete multiple access control policies, select the policies, and click **Delete**. A maximum of 1000 access control policies can be deleted at a time.

**Step 6** Click **Yes**.

----End
9 Environment Management

9.1 Creating Environments and Environment Variables

Scenarios

An API can be called in different environments, such as the production, test, and development environments. RELEASE is the default environment provided by API Gateway. Environment variables can be defined 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 defined during API creation, corresponding variables and their values must be created. For example, variable Path is defined for an API, and two variables with the same name are respectively created and assigned values /Stage/test and /Stage/AA in environments 1 and 2. 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 9-1 Application of environment variables

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

An API group has been created.

Creating an Environment

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > Environments.

**Step 5** Click Create Environment, and set the parameters listed in Table 9-1.

![Figure 9-2 Creating an environment](image)

**Table 9-1 Environment information**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Environment name</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the environment</td>
</tr>
</tbody>
</table>

**Step 6** Click OK.

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

---End

Accessing an Environment

The default RELEASE environment can be accessed using a RESTful API. To access other environments, add the X-Stage header to the request to specify an environment name. For example, to access the DEVELOP environment, add **X-Stage: DEVELOP** to the request header.

Creating an Environment Variable

**Step 1** Log in to the management console.
Step 2  Click in the upper left corner to select a region.

Step 3  Choose Service List > Application > API Gateway.

Step 4  In the navigation pane, choose API Publishing > API Groups.

Step 5  Create a variable in either of the following two ways:

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

Step 7  Set the parameters listed in Table 9-2.

Figure 9-3 Creating an environment variable

Table 9-2 Parameters for creating an environment variable

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the variable you want to create. The name must be the same as that of the variable defined for the API.</td>
</tr>
<tr>
<td>Value</td>
<td>The path to be used in the selected environment.</td>
</tr>
</tbody>
</table>

Step 8  Click OK.

**NOTE**

If the variable is not needed, click Delete in the same row to delete it.

---End

**Follow-Up Operations**

After creating an environment and variable, publish APIs in the environment for API callers to invoke.
Creating an Environment and Variable Using REST APIs

Alternatively, create an environment and variable by using REST APIs provided by API Gateway. For more information, see the following references.

Creating an Environment

Creating an Environment Variable

FAQs About Environment and Variable Creation

Can I Invoke Different Backend Services by Publishing an API in Different Environments?

9.2 Deleting an Environment

Scenarios

Delete an environment if it is not needed to provide services.

Prerequisites

An environment has been created.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner to select a region.

Step 3 Choose Service List > Application > API Gateway.

Step 4 In the navigation pane, choose API Publishing > Environments.

Step 5 In the Operation column of the environment you want to delete, click Delete.

NOTE

An environment can be deleted only if no APIs have been published in the environment.

Step 6 Click Yes.

Deleting an Environment Using a REST API

Alternatively, delete an environment by using a REST API provided by API Gateway. For more information, see the following reference.

Deleting an Environment
10 Signature Key Management

10.1 Creating and Using a Signature Key

Scenarios

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

Each signature key consists of a key and secret, and takes effect only after being bound to APIs. When an API bound with a signature key is called, API Gateway adds signature information to the API requests. The backend service of the API signs the requests 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 be bound with only one signature key in the same environment, and each signature key can be bound to multiple APIs.

Process Flow

1. Create a signature key on the API Gateway console.
2. Bind the signature key to an API. API Gateway signs each request sent to call the 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.
Creating a Signature Key

Step 1  Log in to the management console.

Step 2  Click in the upper left corner to select a region.

Step 3  Choose Service List > Application > API Gateway.

Step 4  In the navigation pane, choose API Publishing > Signature Keys.

Step 5  Click Create Signature Key.

Step 6  In the Create Signature Key dialog box, set the parameters listed in Table 10-1.

Table 10-1 Parameters for creating a signature key

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Signature key name.</td>
</tr>
<tr>
<td>Key</td>
<td>Applies with Secret to form a signature key pair.</td>
</tr>
<tr>
<td>Secret</td>
<td>Applies with Key to form a signature key pair.</td>
</tr>
</tbody>
</table>
Step 7  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 in either of the following two ways:

  ●  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 filter the desired API.

Step 5  Select the API and click OK.

Abbreviation

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 Signature Algorithm, and check whether the backend signature is consistent with the signature in the Authorization header of the API request.

Creating a Signature Key Using a REST API

Alternatively, create a signature key by using a REST API provided by API Gateway. For more information, see the following reference.

Creating a Signature Key

10.2 Deleting a Signature Key

Scenarios

Delete a signature key if it is not needed to provide services.

Prerequisites

A signature key has been created.

Procedure

Step 1  Log in to the management console.

Step 2  Click in the upper left corner to select a region.

Step 3  Choose Service List > Application > API Gateway.
Step 4  In the navigation pane, choose API Publishing > Signature Keys.

Step 5  Delete a signature key in either of the following two ways:

- 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 APIs, unbind it from the APIs before deleting it.

Step 6  Click Yes.

--- End

Deleting a Signature Key Using a REST API

Alternatively, delete a signature key by using a REST API provided by API Gateway. For more information, see the following reference.

**Deleting a Signature Key**
11 VPC Channel Management

11.1 Creating a VPC Channel

Scenarios

Create a VPC channel for accessing resources in a VPC and opening up capabilities of the backend services deployed in the VPC. A VPC channel forwards API requests to different servers for load balancing.

After creating a VPC channel, you can configure it for an API with an HTTP/HTTPS backend service.

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 11-1** Accessing the ECSs in a VPC channel through API Gateway

![Diagram](image)

**NOTE**

You can create a maximum of 30 VPC channels.

Prerequisites

ECSs have been created.
Creating a Fast Channel

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > VPC Channels.

**Step 5** Click Create Fast Channel, and set the parameters listed in Table 11-1.

![Figure 11-2 Creating a fast channel](image)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>VPC channel name</td>
</tr>
<tr>
<td>Port</td>
<td>Host port of the VPC channel</td>
</tr>
<tr>
<td></td>
<td>Range: 1 - 65535</td>
</tr>
</tbody>
</table>

![Table 11-1 Parameters for creating a VPC channel](image)
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distribution</td>
<td>Determines to which host requests will be sent.</td>
</tr>
<tr>
<td>Algorithm</td>
<td>The following distribution algorithms are available:</td>
</tr>
<tr>
<td></td>
<td>• WRR: weighted round robin</td>
</tr>
<tr>
<td></td>
<td>• WLC: weighted least connection</td>
</tr>
<tr>
<td></td>
<td>• SH: source hashing</td>
</tr>
<tr>
<td></td>
<td>• URI hashing</td>
</tr>
<tr>
<td>Protocol</td>
<td>Used to perform health checks on ECSs associated with the VPC channel. Options:</td>
</tr>
<tr>
<td></td>
<td>• TCP</td>
</tr>
<tr>
<td></td>
<td>• HTTP</td>
</tr>
<tr>
<td></td>
<td>Default value: TCP</td>
</tr>
<tr>
<td>Path</td>
<td>Destination path for health checks</td>
</tr>
<tr>
<td></td>
<td>Set this parameter only when Protocol is set to HTTP.</td>
</tr>
<tr>
<td>Check Port</td>
<td>Destination port for health checks</td>
</tr>
<tr>
<td></td>
<td>By default, the host port of the VPC channel is used.</td>
</tr>
<tr>
<td>Healthy Threshold</td>
<td>Number ((x)) of consecutive successful checks required for an ECS to be considered healthy. (x) refers to the healthy threshold.</td>
</tr>
<tr>
<td>Unhealthy Threshold</td>
<td>Number ((x)) of consecutive failed checks required for an ECS to be considered unhealthy. (x) refers to the unhealthy threshold.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Timeout used to determine whether a health check fails. Unit: s.</td>
</tr>
<tr>
<td>Interval</td>
<td>Interval between consecutive checks. Unit: s.</td>
</tr>
<tr>
<td>Response Codes</td>
<td>Response codes for determining a successful HTTP response</td>
</tr>
<tr>
<td></td>
<td>Set this parameter only when Protocol is set to HTTP.</td>
</tr>
</tbody>
</table>

**Step 6**  Click Next.

**Step 7**  Click Select ECS.

**Step 8**  Select the ECSs you want to add, and click OK.

**NOTE**

To ensure successful health check and service availability, the 100.125.0.0/16 segment must have been configured for security groups of the backend ECSs in the inbound direction.

**Step 9**  Click Finish.

--- End
Creating a VPC Channel Using a REST API

Alternatively, create a VPC channel by using a REST API provided by API Gateway. For more information, see the following reference.

Creating a VPC Channel

Follow-Up Operations

Create an API to open up capabilities of backend services deployed in a VPC for load balancing.

11.2 Deleting a VPC Channel

Scenarios

Delete a VPC channel if it is not needed.

**NOTE**

VPC channels that are used by published APIs cannot be deleted.

Prerequisites

A VPC channel has been created.

Procedure

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > VPC Channels.

**Step 5** Delete a VPC channel in either of the following two ways:

- 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 6** Click Yes.

**End**

Deleting a VPC Channel Using a REST API

Alternatively, delete a VPC channel by using a REST API provided by API Gateway. For more information, see the following reference.

Deleting a VPC Channel
11.3 Editing Health Check Configurations

Scenarios

Modify the health check configurations of a VPC channel as necessary.

Prerequisites

A VPC channel has been created.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner to select a region.

Step 3 Choose Service List > Application > API Gateway.

Step 4 In the navigation pane, choose API Publishing > VPC Channels.

Step 5 Click the name of the target VPC channel.

Step 6 Click the Health Check tab.

Step 7 Click Edit Health Check.

Step 8 In the Edit Health Check Configuration dialog box, modify the parameters listed in Table 11-2.

Table 11-2 Health check configurations

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>Used to perform health checks on ECSs associated with the VPC channel. Options:</td>
</tr>
<tr>
<td></td>
<td>- TCP</td>
</tr>
<tr>
<td></td>
<td>- HTTP</td>
</tr>
<tr>
<td></td>
<td>Default value: TCP</td>
</tr>
<tr>
<td>Path</td>
<td>Destination path for health checks</td>
</tr>
<tr>
<td></td>
<td>Set this parameter only when Protocol is set to HTTP.</td>
</tr>
<tr>
<td>Check Port</td>
<td>Destination port for health checks</td>
</tr>
<tr>
<td></td>
<td>By default, the host port of the VPC channel is used.</td>
</tr>
<tr>
<td>Healthy Threshold</td>
<td>Number (x) of consecutive successful checks required for an ECS to be considered healthy. x refers to the healthy threshold.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unhealthy Threshold</td>
<td>Number ((x)) of consecutive failed checks required for an ECS to be</td>
</tr>
<tr>
<td></td>
<td>considered unhealthy. (x) refers to the unhealthy threshold.</td>
</tr>
<tr>
<td></td>
<td>Range: 2 - 10. Default value: 5.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Timeout used to determine whether a health check fails. Unit: s.</td>
</tr>
<tr>
<td>Interval</td>
<td>Interval between consecutive checks. Unit: s.</td>
</tr>
<tr>
<td></td>
<td>Range: 5 - 300. Default value: 10.</td>
</tr>
<tr>
<td>Response Codes</td>
<td>HTTP codes used to check for an HTTP response from a target. Set this</td>
</tr>
<tr>
<td></td>
<td>parameter only when Protocol is set to HTTP.</td>
</tr>
</tbody>
</table>

**Step 9** Click OK.

---End

### 11.4 Editing ECS Configurations of a VPC Channel

#### Scenarios

After creating a VPC channel, you can add or remove ECSs and edit ECS weights for the VPC channel as necessary.

#### Prerequisites

A VPC channel has been created.

#### Procedure

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner to select a region.

**Step 3** Choose Service List > Application > API Gateway.

**Step 4** In the navigation pane, choose API Publishing > VPC Channels.

**Step 5** Click the name of the target VPC channel.

**Step 6** Click the ECSs tab.

**Step 7** Add or delete ECSs and edit ECS weights as required.

- Adding ECSs
  
  a. Click Select ECS.
  
  b. Select the ECSs you want to add, set ECS weights, and click OK.
NOTE

To ensure successful health check and service availability, the 100.125.0.0/16 segment must have been configured for security groups of the backend ECSs in the inbound direction.

- Removing an ECS
  a. In the Operation column of the ECS you want to remove, click Remove.
  b. Click Yes.
- Editing the weight of an ECS
  a. In the Weight column of the target ECS, click [edit].
  b. Change the weight and click ✓.
- Editing the weights of multiple ECSs
  a. Select the ECSs to be edited, and click Edit Weight.
  b. Change the weights of desired ECSs, and click OK.

----End

Editing ECS Configurations of a VPC Channel by Using a REST API

Alternately, edit ECS configurations of a VPC channel by using a REST API provided by API Gateway. For more information, see the following reference.

Adding Backend Instances (ECSs)
12 Auditing

Enabling CTS

If you want to collect, record, or query operation logs of API Gateway in common scenarios such as security analysis, compliance audit, resource tracing, and problem locating, enable CTS first. For details, see Enabling CTS.

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 Querying Real-Time Traces.

Viewing Key Operations

With CTS, you can record operations associated with API Gateway for later query, audit, and backtrack operations. For details, see Key Operations on API Gateway.

Disabling CTS

Disable CTS by following the procedure in Deleting a Tracker.
13 Monitoring

13.1 API Gateway Metrics

Table 13-1 API Gateway metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requests</td>
<td>Total number of times an API is called</td>
</tr>
<tr>
<td>Average latency</td>
<td>Average latency an API is called in a period</td>
</tr>
<tr>
<td>Maximum latency</td>
<td>Maximum latency an API is called in a period</td>
</tr>
<tr>
<td>Upstream throughput</td>
<td>Total incoming traffic of all API requests in a period</td>
</tr>
<tr>
<td>Downstream throughput</td>
<td>Total outgoing traffic of all API responses in a period</td>
</tr>
<tr>
<td>5XX errors</td>
<td>Total number of errors whose code begin with number &quot;5&quot;</td>
</tr>
<tr>
<td>4XX errors</td>
<td>Total number of errors whose code begin with number &quot;4&quot;</td>
</tr>
</tbody>
</table>

13.2 Creating Alarm Rules

Scenarios

You can create alarm rules to track the running status of your APIs and prevent service abnormalities.

An alarm rule consists of the 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 to select a region.

Step 3  Choose Service List > Application > API Gateway.

Step 4  In the navigation pane, choose API Publishing > APIs.

Step 5  Click the name of the target API.

Step 6  On the Monitoring tab page, click View Metric to access the Cloud Eye console. Create alarm rules by following the procedure in Creating Alarm Rules.

----End

13.3 Viewing Metrics

Scenarios

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

Prerequisites

An API group and API have been created.

Procedure

Step 1  Log in to the management console.

Step 2  Click in the upper left corner to select a region.

Step 3  Choose Service List > Application > API Gateway.

Step 4  In the navigation pane, choose API Publishing > APIs.

Step 5  Click the name of the target API.

API metrics are displayed on the Monitoring tab page.

Step 6  Click View Metric to view complete metrics on the Cloud Eye console.

NOTE

The monitoring data is retained for two days. For longer retention periods, save data to an OBS bucket. For details, see Transferring Metric Data to OBS.

----End
## Change History

### Table A-1 Change history

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
</table>
| 2019-06-28 | • Added support for publishing and deleting APIs in batches, taking multiple APIs offline, and granting access to multiple APIs to an app.  
• Added support for deleting request throttling and access control policies in batches.  
• Added support for unbinding a request throttling policy or an access control policy from multiple APIs. |
| 2019-05-29 | Added section "Permissions Management".                                      |
| 2019-05-09 | Optimized the document structure, combining closely-related chapters to meet users' needs in specific scenarios. |
| 2018-12-13 | Added section "Exporting APIs".                                              |
| 2018-11-09 | Added section "Importing APIs".                                              |
| 2018-10-15 | • Added descriptions about the **Max. IP Address Requests** parameter that controls the maximum number of times a single IP address can access an API in section "Creating a Request Throttling Policy".  
• Added section "Adding an Excluded Tenant" in chapter "Request Throttling". |
| 2018-08-31 | Added descriptions about switching API versions in section "Publication History". |
| 2018-07-26 | • Added chapter "Access Control" to describe how to specify IP addresses or IP address ranges that are allowed or not allowed to access an API.  
• Added support for configuring and mapping input and backend parameters. |
<p>| 2018-07-16 | Added chapter &quot;Experiencing Demo&quot; to demonstrate how to use API Gateway.     |</p>
<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018-06-29</td>
<td>• Added support for configuring CORS and constant parameters for an API.</td>
</tr>
<tr>
<td></td>
<td>• Added section &quot;Creating an API with the OPTIONS Method&quot; in chapter &quot;API Management&quot;.</td>
</tr>
<tr>
<td>2018-05-30</td>
<td>Added chapter &quot;VPC Channel Management&quot;.</td>
</tr>
<tr>
<td>2018-05-02</td>
<td>• Added chapter &quot;API Gateway and Other Services&quot;.</td>
</tr>
<tr>
<td></td>
<td>• Added section &quot;Binding a Domain Name&quot; in chapter &quot;API Group Management&quot;.</td>
</tr>
<tr>
<td></td>
<td>• Added section &quot;Publication History&quot; in chapter &quot;API Management&quot;.</td>
</tr>
<tr>
<td></td>
<td>• Added chapter &quot;Monitoring&quot;.</td>
</tr>
<tr>
<td></td>
<td>• Added chapter &quot;Auditing&quot;.</td>
</tr>
<tr>
<td>2018-04-04</td>
<td>Adjusted the quotas of APIs, API groups, request throttling policies, environments, and signature keys.</td>
</tr>
<tr>
<td>2018-03-09</td>
<td>Added section &quot;Enabling CTS&quot;.</td>
</tr>
<tr>
<td>2018-01-31</td>
<td>• Added section &quot;Creating an Environment Variable&quot; in chapter &quot;API Group Management&quot;.</td>
</tr>
<tr>
<td></td>
<td>• Added chapter &quot;Signature Key Management&quot;.</td>
</tr>
<tr>
<td>2017-12-25</td>
<td>This issue is the first official release.</td>
</tr>
</tbody>
</table>