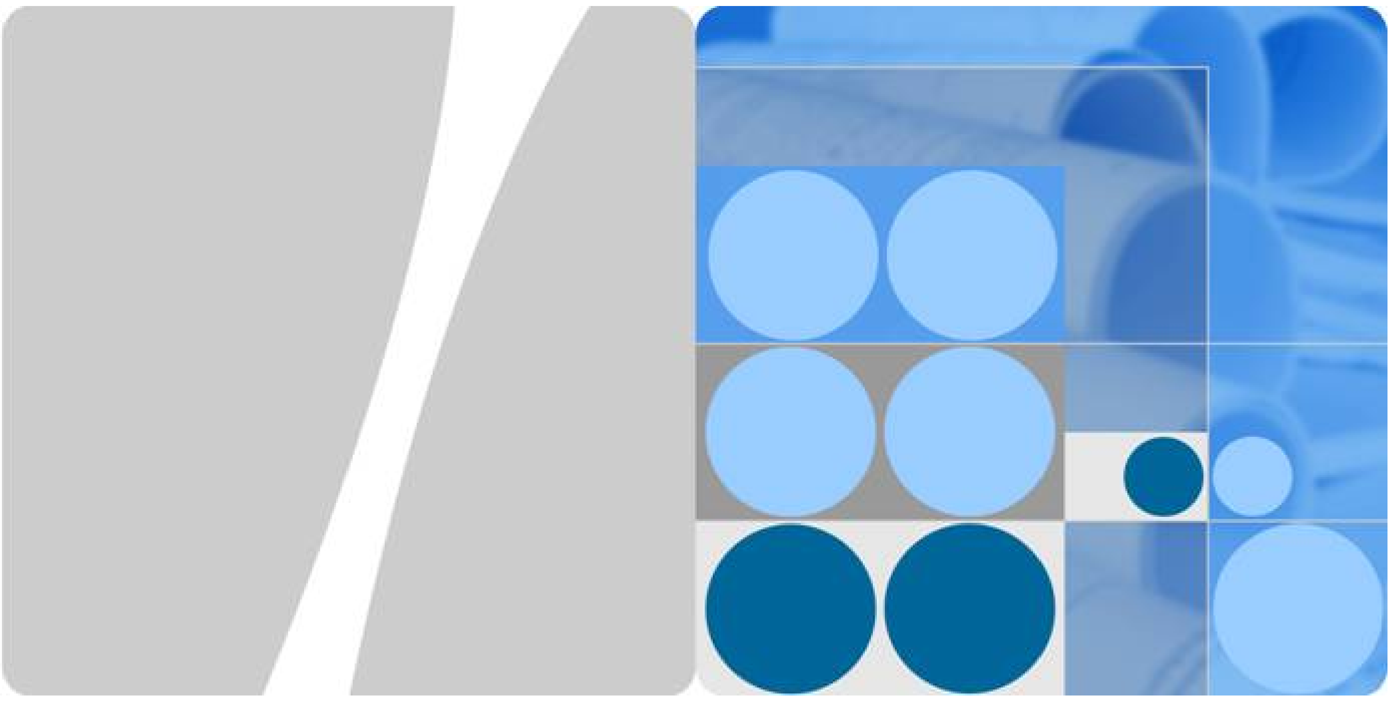


API Gateway

Best Practices

Date 2023-04-30

Contents

1 Selectively Exposing CCE Workloads

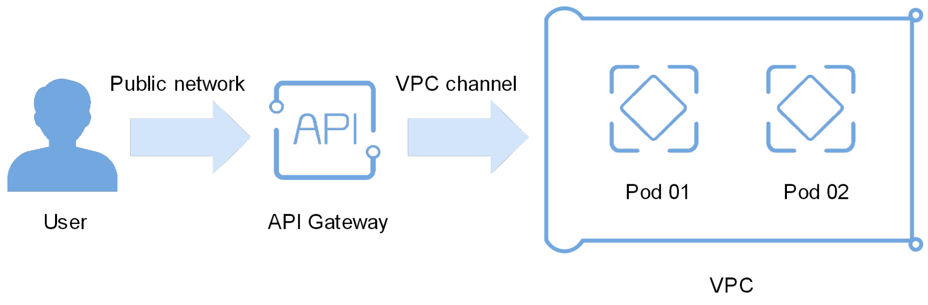
Overview

You can use APIG to selectively expose your workloads and microservices in Cloud Container Engine (CCE). Using APIG to expose containerized applications has the following benefits:

- You do not need to set elastic IP addresses, and this reduces network bandwidth costs.

  You can set up a VPC channel to access workloads in CCE.
- You can choose an authentication mode from multiple options to ensure access security.
- You can configure a request throttling policy to ensure secure access to your backend service.
- You can configure multiple pods for each workload for load balancing, optimizing resource utilization and increasing system reliability.

Figure 1-1 Accessing CCE workloads through APIG



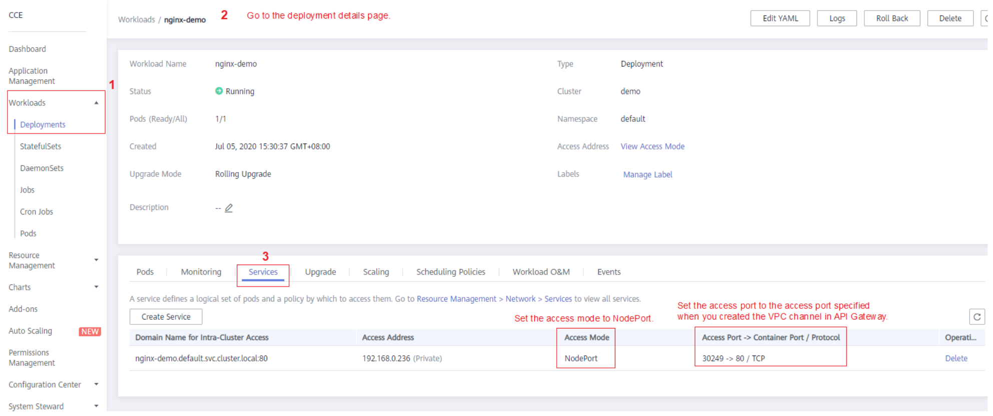
Preparing CCE Workloads

Create a cluster and workload in CCE, and add pods and containers to the workload. For more information, see *CCE User Guide*.

View the workload details on the CCE console, and ensure that the service access mode is **NodePort** or **LoadBalancer**. For details, see section "NodePort" or section "LoadBalancer".

- NodePort access

Figure 1-2 Viewing the access port



Go to the deployment details page.

Workload Name	nginx-demo	Type	Deployment
Status	Running	Cluster	demo
Pods (Ready/All)	1/1	Namespace	default
Created	Jul 05, 2020 15:30:37 GMT+08:00	Access Address	View Access Mode
Upgrade Mode	Rolling Upgrade	Labels	Manage Label
Description	--		

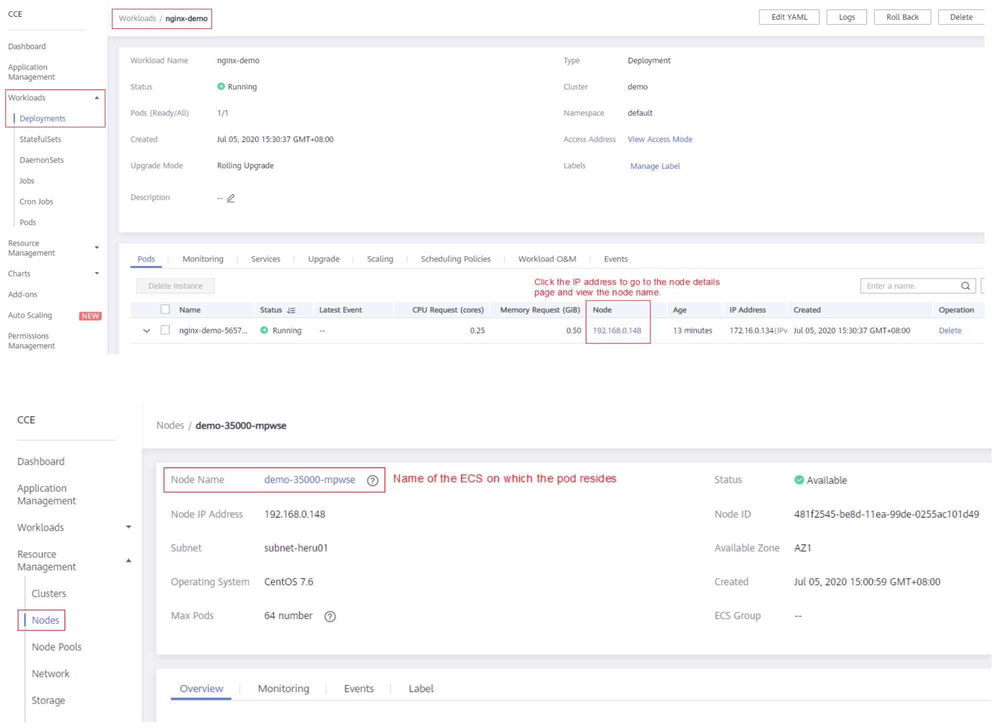
A service defines a logical set of pods and a policy by which to access them. Go to Resource Management > Network > Services to view all services.

Set the access mode to NodePort.

Set the access port to the access port specified when you created the VPC channel in API Gateway.

Domain Name for Intra-Cluster Access	Access Address	Access Mode	Access Port -> Container Port / Protocol	Operati...
nginx-demo.default.svc.cluster.local:80	192.168.0.236 (Private)	NodePort	30249 -> 80 / TCP	Delete

Figure 1-3 Viewing the name of the ECS on which the pod resides



Workload Name	nginx-demo	Type	Deployment
Status	Running	Cluster	demo
Pods (Ready/All)	1/1	Namespace	default
Created	Jul 05, 2020 15:30:37 GMT+08:00	Access Address	View Access Mode
Upgrade Mode	Rolling Upgrade	Labels	Manage Label
Description	--		

Click the IP address to go to the node details page and view the node name.

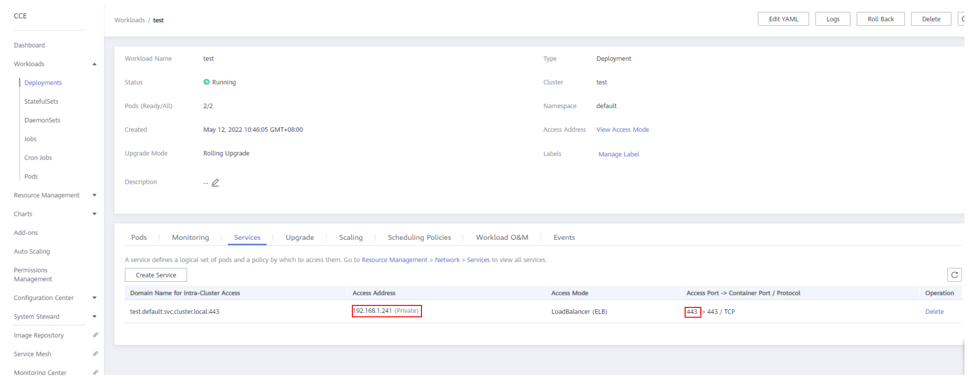
Name	Status	Latest Event	CPU Request (cores)	Memory Request (GiB)	Node	Age	IP Address	Created	Operation
nginx-demo-5657...	Running	--	0.25	0.50	192.168.0.148	13 minutes	172.16.0.134(IPv	Jul 05, 2020 15:30:37 GMT+08:00	Delete

Nodes / demo-35000-mpwse

Name of the ECS on which the pod resides

Node Name	demo-35000-mpwse	Status	Available
Node IP Address	192.168.0.148	Node ID	481f2545-be8d-11ea-99de-0255ac101d49
Subnet	subnet-heru01	Available Zone	AZ1
Operating System	CentOS 7.6	Created	Jul 05, 2020 15:00:59 GMT+08:00
Max Pods	64 number	ECS Group	--

- LoadBalancer access



Workload Name	test	Type	Deployment
Status	Running	Cluster	test
Pods (Ready/All)	2/2	Namespace	default
Created	May 12, 2022 10:46:05 GMT+08:00	Access Address	View Access Mode
Upgrade Mode	Rolling Upgrade	Labels	Manage Label
Description	--		

A service defines a logical set of pods and a policy by which to access them. Go to Resource Management > Network > Services to view all services.

Domain Name for Intra-Cluster Access	Access Address	Access Mode	Access Port -> Container Port / Protocol	Operation
test.default.svc.cluster.local:443	192.168.1.241 (Private)	LoadBalancer (ELB)	443 -> 443 / TCP	Delete

Creating a VPC Channel

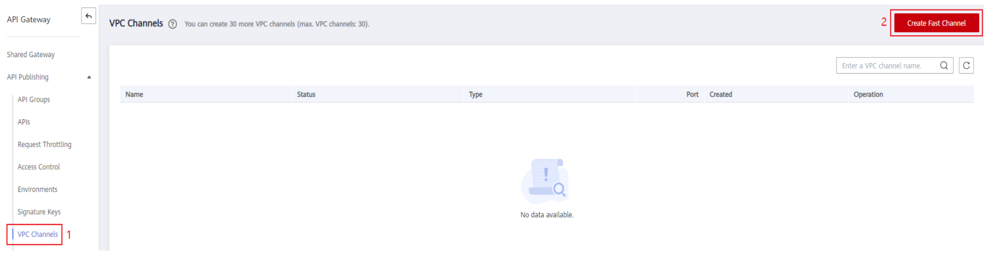
If the access mode of the target CCE workload is **LoadBalancer**, skip this procedure and go to **Opening an API**.

Step 1 Log in to the management console, select a region in the upper left corner, and choose **Service List** > **Application** > **API Gateway**.

Step 2 Create a VPC channel.

1. On the **VPC Channels** page, click **Create Fast Channel**.

Figure 1-4 VPC channel list

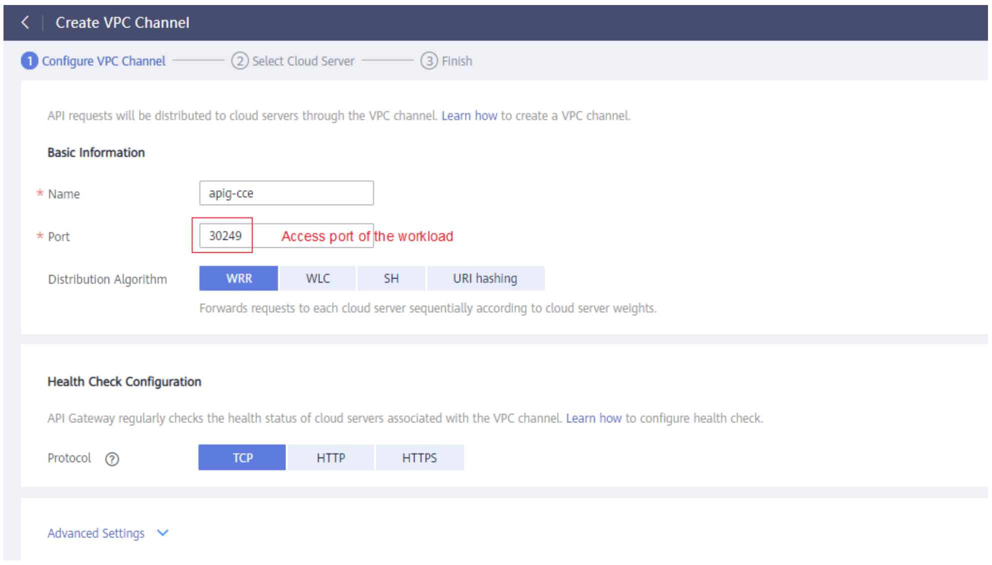


You can create 30 more VPC channels (max. VPC channels: 30).

2. Set the parameters according to the following figure and retain the default values for other parameters.

   For details, see *API Gateway User Guide*.

Figure 1-5 Setting the basic VPC channel information



API requests will be distributed to cloud servers through the VPC channel. Learn how to create a VPC channel.

* Name: apig-cce

* Port: 30249 Access port of the workload

Distribution Algorithm: WRR

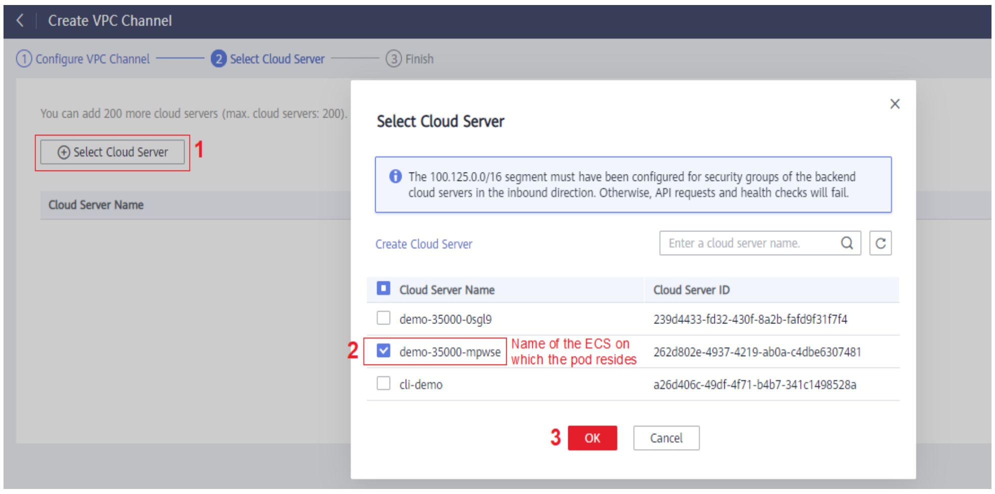
Forwards requests to each cloud server sequentially according to cloud server weights.

API Gateway regularly checks the health status of cloud servers associated with the VPC channel. Learn how to configure health check.

Protocol: TCP

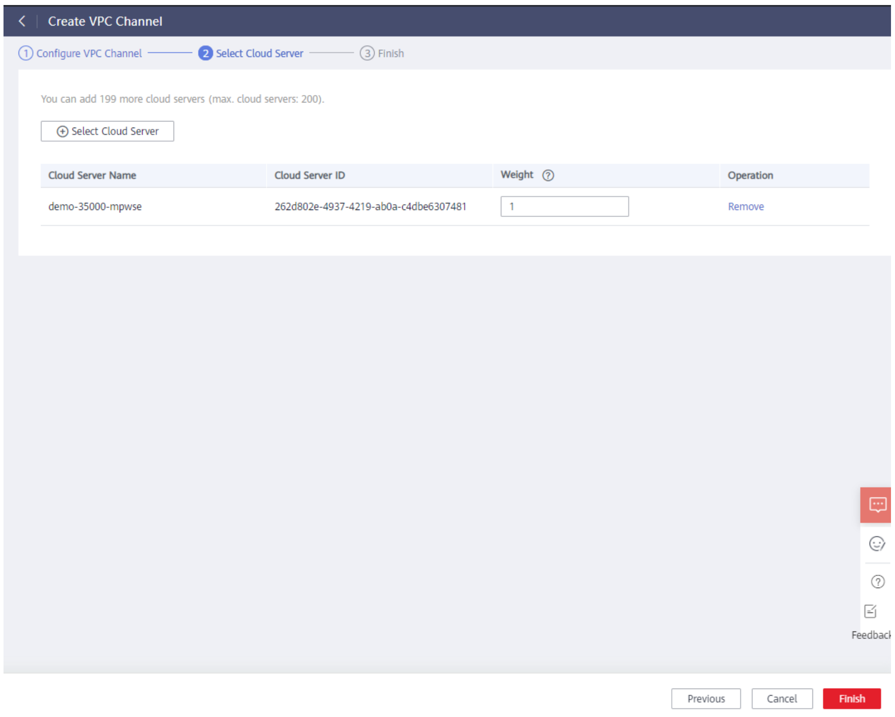
Step 3 Add the node that contains the CCE workload you want to access through APIG.

You can add multiple nodes for load balancing.


Create VPC Channel
1 Configure VPC Channel
2 Select Cloud Server
3 Finish
You can add 200 more cloud servers (max. cloud servers: 200).
Select Cloud Server
1
Cloud Server Name
Select Cloud Server
The 100.125.0.0/16 segment must have been configured for security groups of the backend cloud servers in the inbound direction. Otherwise, API requests and health checks will fail.
Create Cloud Server
Enter a cloud server name.
2
Name of the ECS on which the pod resides
3
OK
Cancel

Cloud Server Name	Cloud Server ID
demo-35000-0sgl9	239d4433-fd32-430f-8a2b-fafd9f31f7f4
demo-35000-mpwse	262d802e-4937-4219-ab0a-c4dbe6307481
cli-demo	a26d406c-49df-4f71-b4b7-341c1498528a

Step 4 Click **Finish**.



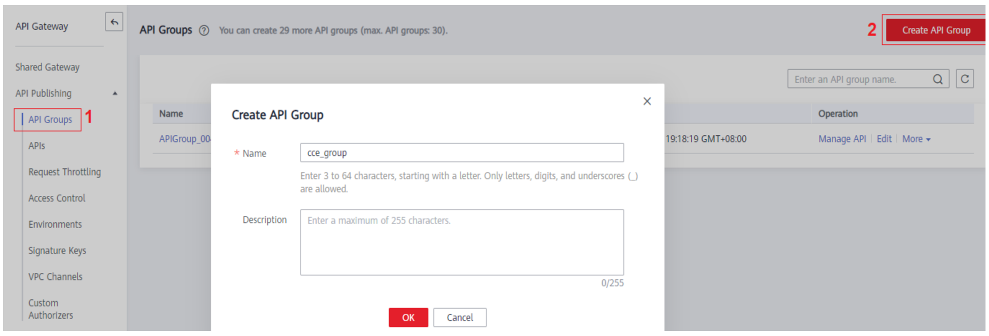
Cloud Server Name	Cloud Server ID	Weight	Operation
demo-35000-mpwse	262d802e-4937-4219-ab0a-c4dbe6307481	1	Remove

----End

Opening an API

Step 1 Create an API group, as shown in **Figure 1-6**.

Figure 1-6 Creating an API group



API Groups You can create 29 more API groups (max. API groups: 30).

Name		Operation
APIGroup_00	19:18:19 GMT+08:00	Manage API | Edit | More

Create API Group

* Name: cce_group

Enter 3 to 64 characters, starting with a letter. Only letters, digits, and underscores (_) are allowed.

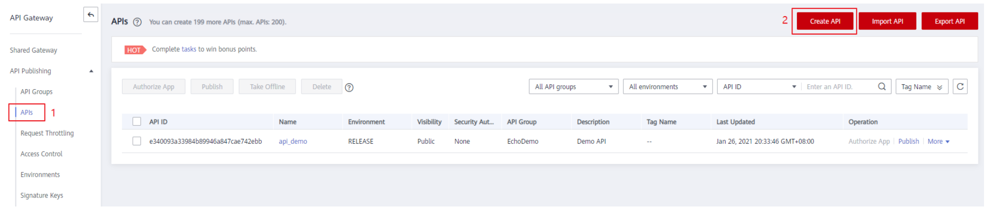
Description: Enter a maximum of 255 characters. 0/255

Step 2 Create an API.

For details, see *API Gateway User Guide*.

1. Click **Create API**.

Figure 1-7 API list



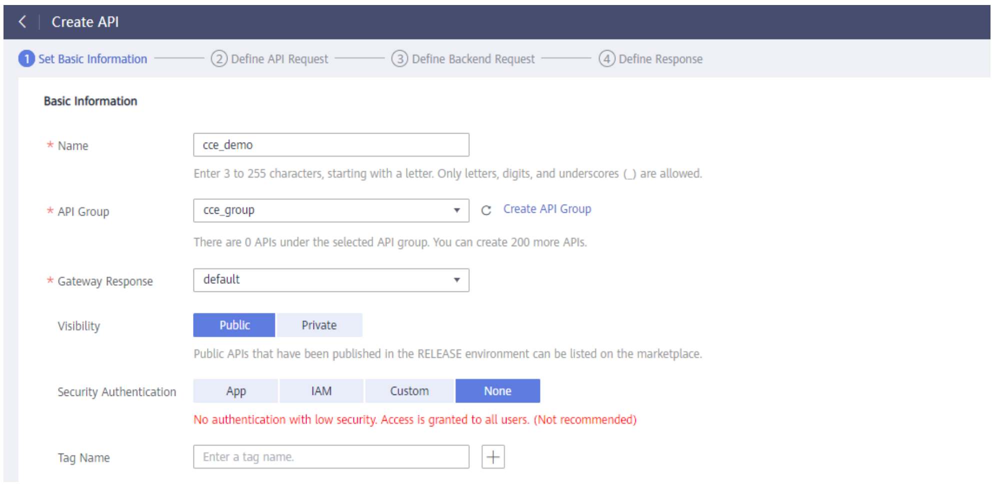
APIs You can create 199 more APIs (max. APIs: 200).

Complete tasks to win bonus points.

API ID	Name	Environment	Visibility	Security Aut...	API Group	Description	Tag Name	Last Updated	Operation
e340093a33984b89946a847cae742ebb	api_demo	RELEASE	Public	None	EchoDemo	Demo API	--	Jan 26, 2021 20:33:46 GMT+08:00	Authorize App | Publish | More

2. Set the basic information of the API.

Figure 1-8 Setting the basic API information



Basic Information

* Name: cce_demo

Enter 3 to 255 characters, starting with a letter. Only letters, digits, and underscores (_) are allowed.

* API Group: cce_group

There are 0 APIs under the selected API group. You can create 200 more APIs.

* Gateway Response: default

Visibility: [✓ Public] Private

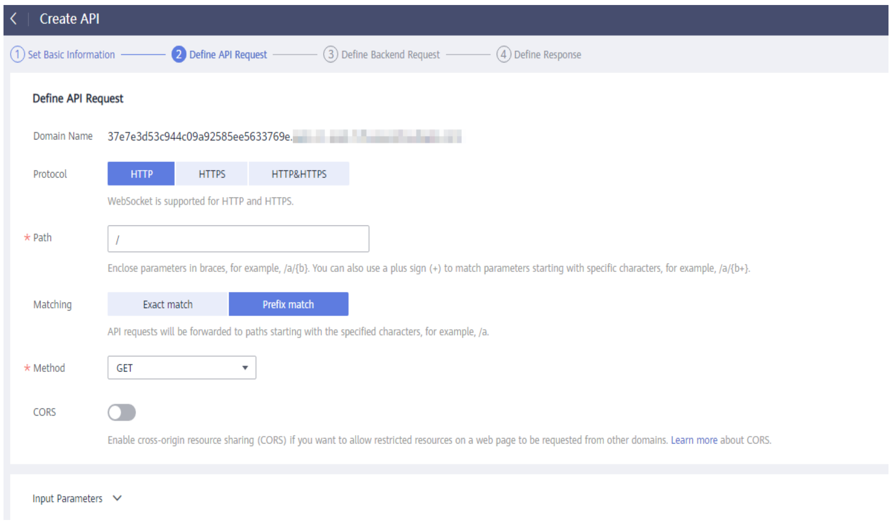
Public APIs that have been published in the RELEASE environment can be listed on the marketplace.

Security Authentication: App IAM Custom [✓ None]

No authentication with low security. Access is granted to all users. (Not recommended)

Tag Name: Enter a tag name.

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



Define API Request

Domain Name 37e7e3d53c944c09a92585ee5633769e.[redacted]

Protocol [✓ HTTP]

WebSocket is supported for HTTP and HTTPS.

* Path /

Enclose parameters in braces, for example, /a/{b}. You can also use a plus sign (+) to match parameters starting with specific characters, for example, /a/{b+}.

Matching [✓ Prefix match]

API requests will be forwarded to paths starting with the specified characters, for example, /a.

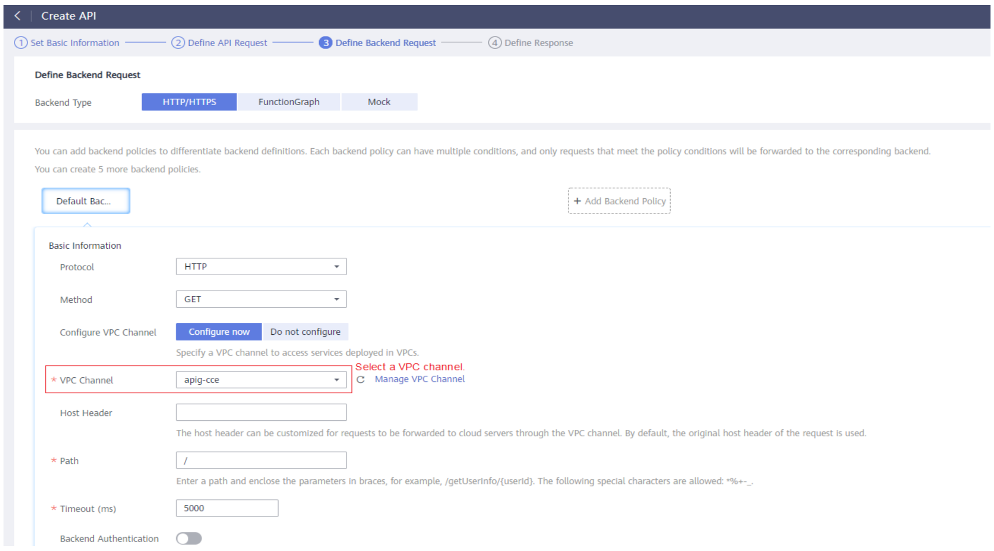
* Method GET

CORS

Enable cross-origin resource sharing (CORS) if you want to allow restricted resources on a web page to be requested from other domains. Learn more about CORS.

4. On the **Define Backend Request** page, set the backend request information.

   If the access mode of the target CCE workload is **NodePort**, select **Configure now**, and select the VPC channel created in **Creating a VPC Channel**. If the access mode is **LoadBalancer**, select **Do not configure**, and enter the **access address and port** of the load balancer. This step uses **NodePort** as an example.



Define Backend Request

Backend Type [✓ HTTP/HTTPS]

You can add backend policies to differentiate backend definitions. Each backend policy can have multiple conditions, and only requests that meet the policy conditions will be forwarded to the corresponding backend.
You can create 5 more backend policies.

Basic Information

Protocol HTTP

Method GET

Configure VPC Channel [✓ Configure now]

Specify a VPC channel to access services deployed in VPCs.

* VPC Channel apig-cce

Host Header

The host header can be customized for requests to be forwarded to cloud servers through the VPC channel. By default, the original host header of the request is used.

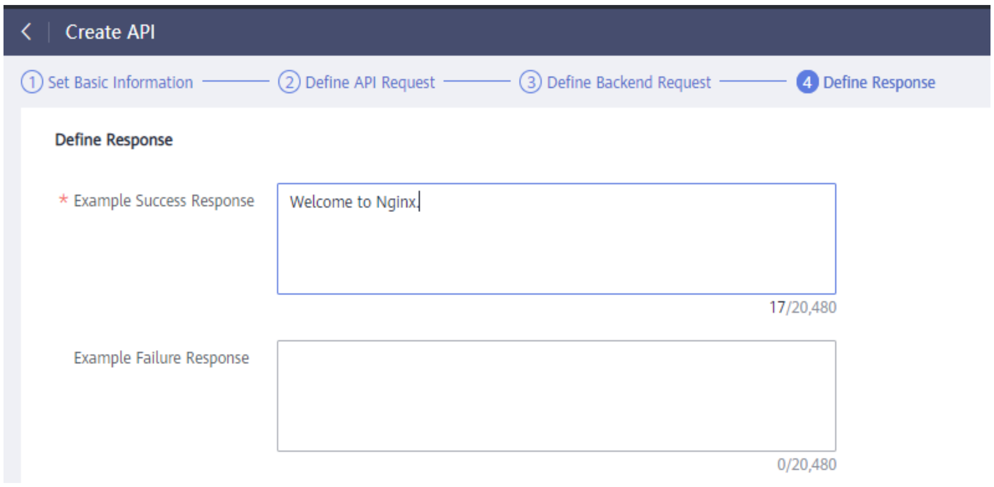
* Path /

Enter a path and enclose the parameters in braces, for example, /getUserInfo/{userId}. The following special characters are allowed: *%+-_.

* Timeout (ms) 5000

Backend Authentication

5. On the **Define Response** page, enter an example success response.



Define Response

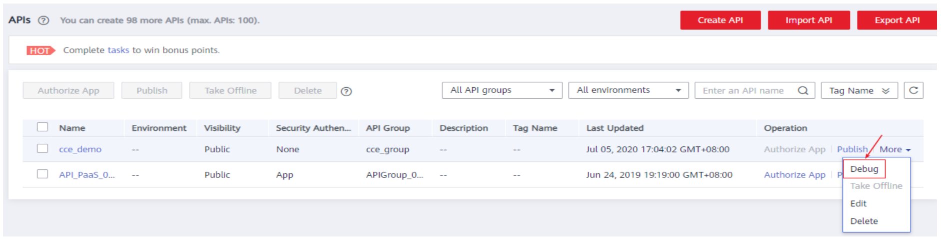
* Example Success Response	Welcome to Nginx (17/20,480)
Example Failure Response	(0/20,480)

6. Click **Finish**.

Step 3 Debug the API.

1. Click **Debug**.

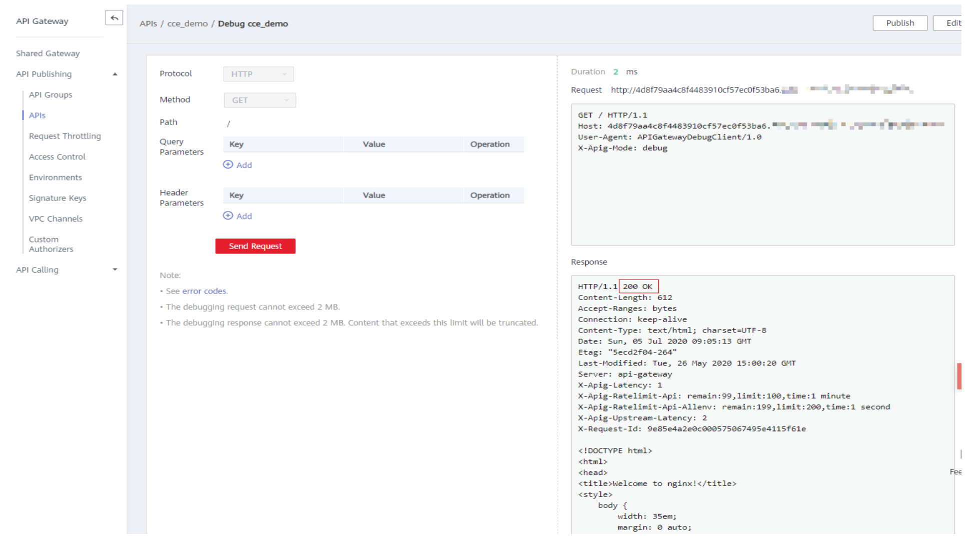
Figure 1-9 API list



Name	Environment	Visibility	Security Authen...	API Group	Description	Tag Name	Last Updated	Operation
cce_demo	--	Public	None	cce_group	--	--	Jul 05, 2020 17:04:02 GMT+08:00	Authorize App | Publish | More
API_PaaS_0...	--	Public	App	APIGroup_0...	--	--	Jun 24, 2019 19:19:00 GMT+08:00	Authorize App | P

2. Debug the API.

Figure 1-10 Debugging the API ("200" indicates that the API is called successfully.)



Protocol HTTP
Method GET
Path /
Query Parameters
Header Parameters

Note:
• See error codes.
• The debugging request cannot exceed 2 MB.
• The debugging response cannot exceed 2 MB. Content that exceeds this limit will be truncated.

Duration 2 ms
Request http://4d8f79aa4c8f4483910cf57ec0f53ba6.[redacted]

```
GET / HTTP/1.1
Host: 4d8f79aa4c8f4483910cf57ec0f53ba6.[redacted]
User-Agent: APIGatewayDebugClient/1.0
X-Apig-Mode: debug
```

Response

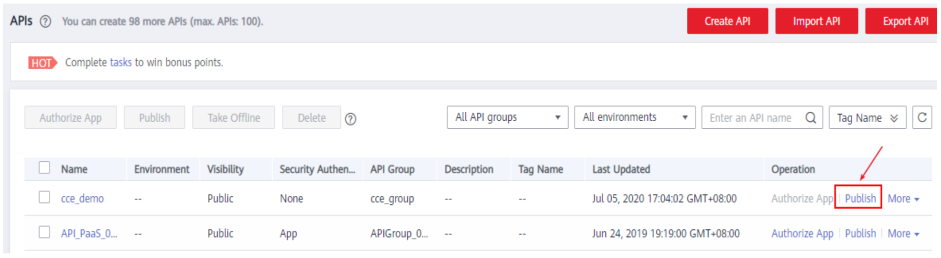
```
HTTP/1.1 200 OK
Content-Length: 612
Accept-Ranges: bytes
Connection: keep-alive
Content-Type: text/html; charset=UTF-8
Date: Sun, 05 Jul 2020 09:05:13 GMT
Etag: "5ecd2f04-264"
Last-Modified: Tue, 26 May 2020 15:00:20 GMT
Server: api-gateway
X-Apig-Latency: 1
X-Apig-Ratelimit-Api: remain:99,limit:100,time:1 minute
X-Apig-Ratelimit-Api-Allenv: remain:199,limit:200,time:1 second
X-Apig-Upstream-Latency: 2
X-Request-Id: 9e85e4a2e0c000575067495e4115f61e

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
```

Step 4 Publish the API.

1. Click **Publish**.

Figure 1-11 API list


APIs You can create 98 more APIs (max. APIs: 100).

Create API | Import API | Export API

HOT Complete tasks to win bonus points.

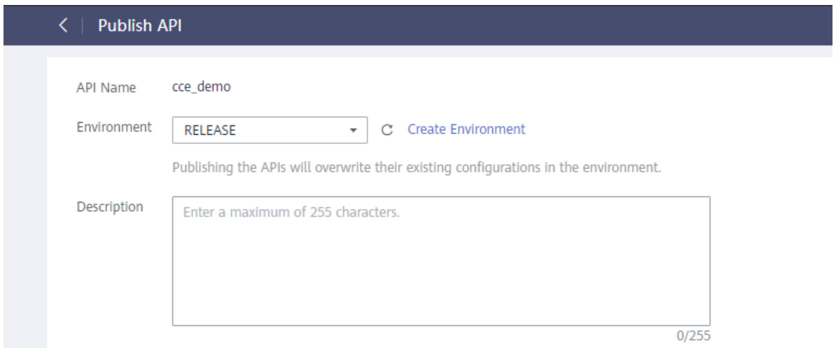
Authorize App | Publish | Take Offline | Delete

All API groups | All environments | Enter an API name | Tag Name

Name	Environment	Visibility	Security Authen...	API Group	Description	Tag Name	Last Updated	Operation
cce_demo	--	Public	None	cce_group	--	--	Jul 05, 2020 17:04:02 GMT+08:00	Authorize App Publish More
API_PaaS_0...	--	Public	App	APIGroup_0...	--	--	Jun 24, 2019 19:19:00 GMT+08:00	Authorize App Publish More

2. Enter a description.

Figure 1-12 Publishing an API


Publish API

API Name: cce_demo

Environment: RELEASE Create Environment

Publishing the APIs will overwrite their existing configurations in the environment.

Description: Enter a maximum of 255 characters.

0/255

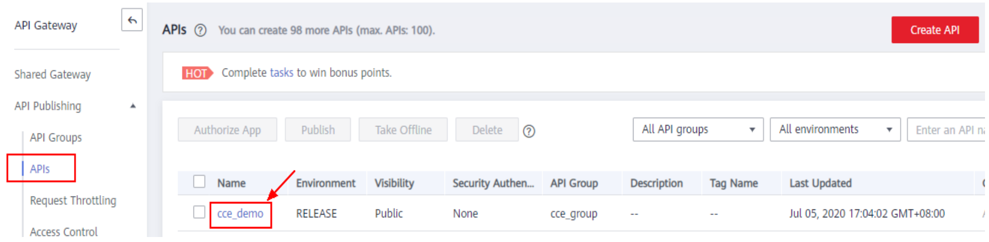
----End

Calling the API

Step 1 In the API list, click the API you created, and copy the URL on the displayed API details page.

1. Go to the API details page.

Figure 1-13 Clicking the name of an API



APIs You can create 98 more APIs (max. APIs: 100).

Create API

HOT Complete tasks to win bonus points.

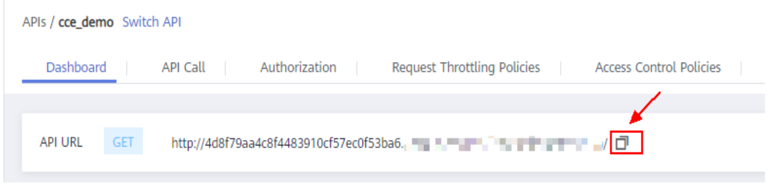
Authorize App | Publish | Take Offline | Delete

All API groups | All environments | Enter an API na

Name	Environment	Visibility	Security Authen...	API Group	Description	Tag Name	Last Updated
cce_demo	RELEASE	Public	None	cce_group	--	--	Jul 05, 2020 17:04:02 GMT+08:00

2. Copy the URL on the displayed API details page.

Figure 1-14 Copying the URL

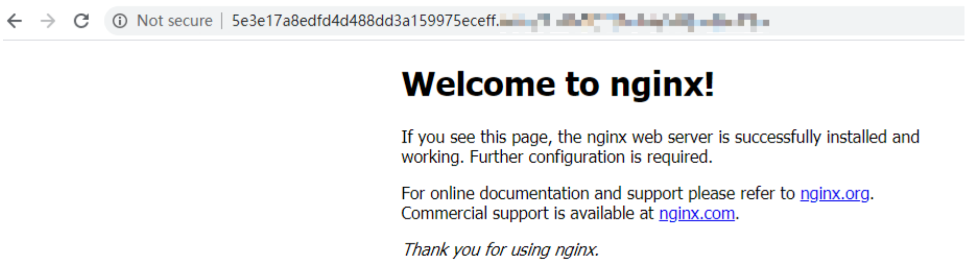

APIs / cce_demo Switch API

Dashboard | API Call | Authorization | Request Throttling Policies | Access Control Policies

API URL GET http://4d8f79aa4c8f4483910cf57ec0f53ba6.[redacted]/

Step 2 Paste the URL to the address bar of a browser. The following page will be displayed if the API request is successful.

To limit the number of API calls that will be received within a specific period, create a request throttling policy and bind it to the API. For more information, see *API Gateway User Guide*.


Not secure | 5e3e17a8edfd4d488dd3a159975eceff.[redacted]

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

----End

2 Selectively Exposing Service Capabilities of a Data Center

The backend services of APIG can be deployed in the following modes:

- Deployed in a VPC and accessible only using private IP addresses.

  You can create a VPC channel on APIG to enable network routing between APIG and the VPC.
- Deployed on the public network and accessible using a public IP address.
- Deployed in an on-premises data center and not accessible using a public IP address.

  If you use a dedicated API gateway, you can set up a connection between your on-premises data center and the gateway.

This section describes the precautions for using APIG to selectively expose APIs of backend services deployed in a local data center.

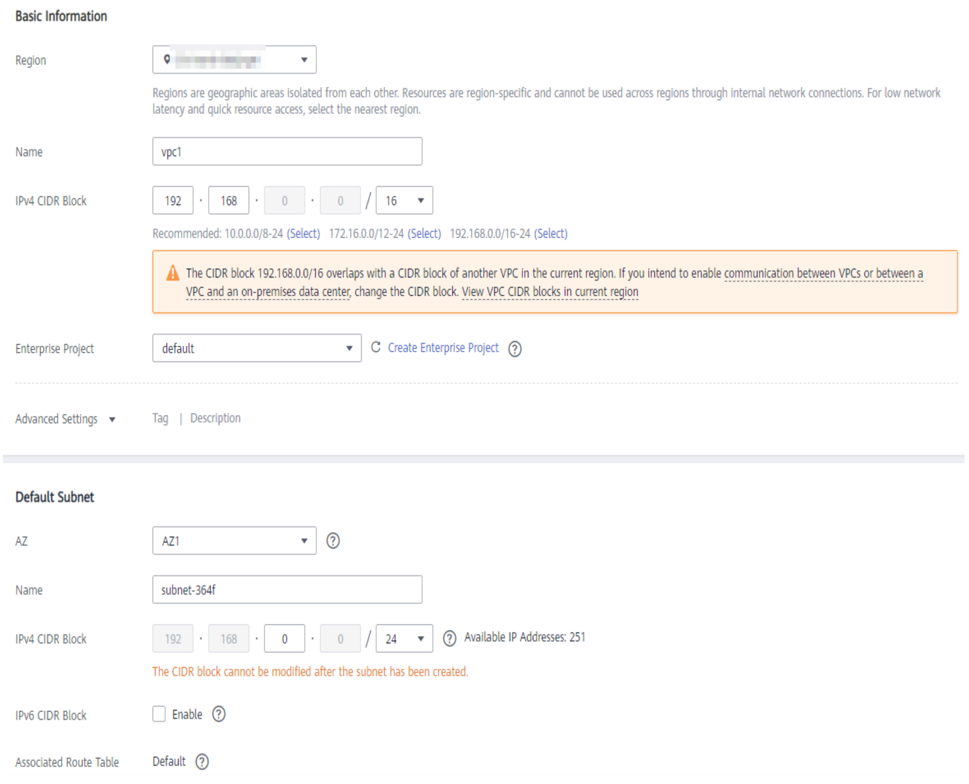
Connecting a Data Center to APIG

Step 1 Create a VPC.

For details, see the section "Creating a VPC" in the *Virtual Private Cloud User Guide*.

To allow APIG to access services in your on-premises data center, bind a VPC to your dedicated gateway, and establish a connection between the data center and VPC.

Figure 2-1 Creating a VPC



Basic Information

Region	[redacted]
	Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.
Name	vpc1
IPv4 CIDR Block	192 . 168 . 0 . 0 / 16
	Recommended: 10.0.0.0/8-24 (Select) 172.16.0.0/12-24 (Select) 192.168.0.0/16-24 (Select)
	The CIDR block 192.168.0.0/16 overlaps with a CIDR block of another VPC in the current region. If you intend to enable communication between VPCs or between a VPC and an on-premises data center, change the CIDR block. View VPC CIDR blocks in current region
Enterprise Project	default
Advanced Settings	Tag | Description

Default Subnet

AZ	AZ1
Name	subnet-364f
IPv4 CIDR Block	192 . 168 . 0 . 0 / 24 Available IP Addresses: 251
	The CIDR block cannot be modified after the subnet has been created.
IPv6 CIDR Block	Enable
Associated Route Table	Default

NOTE

- Specify a subnet for your dedicated gateway.
- A connection can be used to connect a local data center to only one VPC. You are advised to bind the same VPC to all your cloud resources to reduce costs.
- If a VPC already exists, you do not need to create a new one.

Step 2 Buy a dedicated API gateway.

For details, see section "Buying a Gateway" in the *API Gateway User Guide*.

Step 3 Enable Direct Connect by referring to the *Direct Connect User Guide*.

1. Create a connection.

   Apply for a connection from your account manager. If you do not have an account manager, contact technical support.

2. Create a virtual gateway.

   The virtual gateway is a logical gateway for accessing the VPC bound to the dedicated gateway.

   NOTE

   Select the subnet that the dedicated gateway uses, to connect to the VPC. For details about the subnet, go to the gateway details page.

3. Create a virtual interface.

   The virtual interface links the connection with the virtual gateway, enabling connectivity between the connection and the VPC of the dedicated gateway.

   Configure the remote gateway and remote subnet as the gateway and subnet for accessing the open API of your on-premises data center. For example, if the API calling address of your data center is **http://192.168.0.25:80/***{URI}*, configure the remote gateway and remote subnet as those of **192.168.0.25**.

Step 4 Verify the network connectivity.

Create another pay-per-use ECS and select the same VPC, subnet, and security group as the dedicated gateway. If the data center can connect to the ECS, the data center can also connect to the dedicated gateway.

----End

Exposing APIs with the Dedicated Gateway

After you connect the data center to the dedicated gateway, you can expose APIs using the gateway. For details, see "Getting Started" > "Opening APIs" in the *API Gateway User Guide*.

When creating an API, specify the backend address as the API calling address of your data center.

3 Developing a Custom Authorizer with FunctionGraph

Overview

In addition to IAM and app authentication, APIG also supports custom authentication with your own authentication system, which can better adapt to your business capabilities.

Custom authentication is implemented using the FunctionGraph service. You can create a FunctionGraph function so that APIG can invoke it to authenticate requests for your API. This section uses basic authentication as an example to describe how to implement custom authentication with FunctionGraph.

Developing a Custom Authentication Function

Create a function on the FunctionGraph console by referring to section "Creating a Function for Frontend Custom Authentication" in the *Developer Guide*.

Specify the runtime as Python 3.6.

Table 3-1 Function configuration

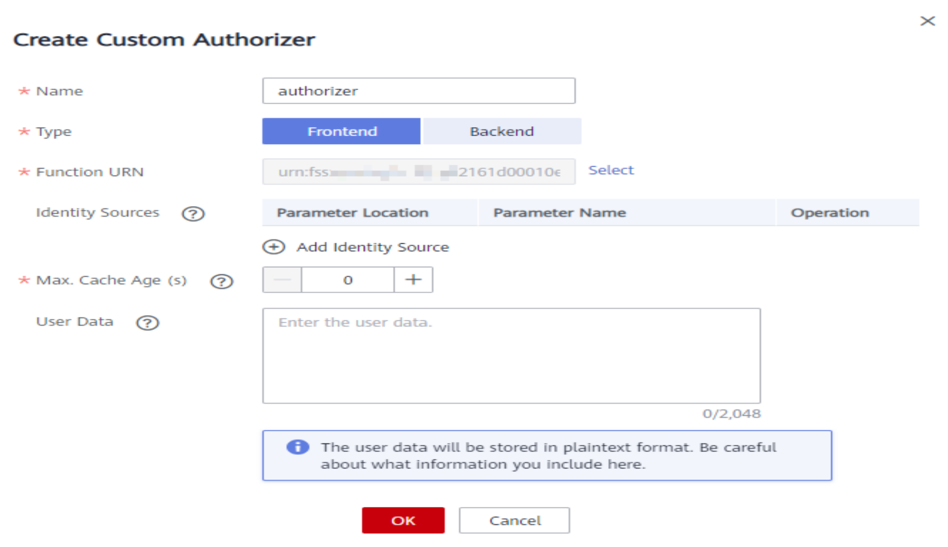
Parameter	Description
Function Type	Default: **Event Function**
Region	Select the same region as that of APIG.
Function Name	Enter a name that conforms to specific rules to facilitate search.
Agency	An agency that delegates FunctionGraph to access other cloud services. For this example, select **Use no agency**.
Enterprise Project	The default option is **default**.
Runtime	Select **Python 3.6**.

On the **Code** tab, copy the following code to **index.py**:

```
# -*- coding:utf-8 -*-
import json
def handler(event, context):
# If the authentication information is correct, the username is returned.
    if event["headers"]["authorization"]=='Basic dXNlcjE6cGFzc3dvcmQ=':
        return {
            'statusCode': 200,
            'body': json.dumps({
                "status":"allow",
                "context":{
                    "user_name":"user1"
                }
            })
        }
    else:
        return {
            'statusCode': 200,
            'body': json.dumps({
                "status":"deny",
                "context":{
                    "code":"1001",
                    "message":"incorrect username or password"
                }
            })
        }
```

Creating a Custom Authorizer

On the APIG console, go to the **Create Custom Authorizer** page, set **Type** to **Frontend**, select the function created in the preceding section, and click **OK**.



Create Custom Authorizer

* Name	authorizer
* Type	[✓ Frontend] Backend
* Function URN	urn:fss[redacted]2161d00010e Select
Identity Sources	Parameter Location / Parameter Name / Operation; Add Identity Source
* Max. Cache Age (s)	0
User Data	Enter the user data. 0/2,048

The user data will be stored in plaintext format. Be careful about what information you include here.

OK Cancel

Creating a Custom Authentication API

Create an API by referring to section "Creating an API" in the *API Gateway User Guide*. Set the authentication mode to **Custom**, and select the custom authorizer created in the preceding section. After modifying the API, publish it.

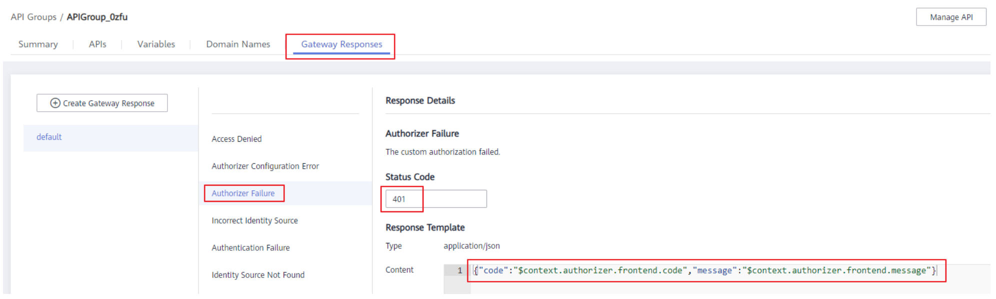
Setting the Error Response

If incorrect authentication information is carried in a request for the API, the response is displayed as follows:

```
{"error_msg":"Incorrect authentication information: frontend authorizer","error_code":"APIG.
0305","request_id":"36e42b3019077c2b720b6fc847733ce9"}
```

To include the **context** field of the function response in the API response result, modify the response template of the API. On the details page of the group to which the API belongs, navigate to the **Gateway Responses** area on the **Gateway Information** tab and click **Edit**. Change the status code to **401**, modify the response template with the following code, and click **OK**:

```
{"code":"$context.authorizer.frontend.code","message":"$context.authorizer.frontend.message"}
```



Authorizer Failure
The custom authorization failed.
Status Code: 401
Response Template
Type: application/json
Content:

```
{"code":"$context.authorizer.frontend.code","message":"$context.authorizer.frontend.message"}
```

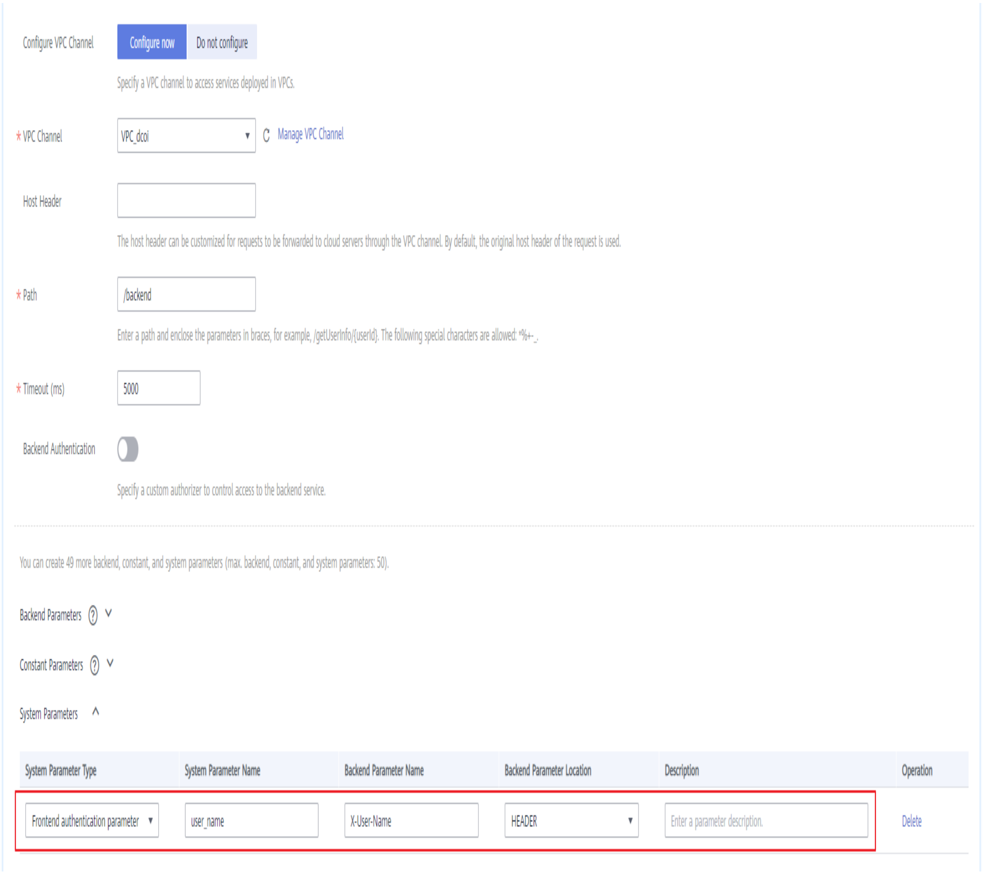
After the modification, if incorrect authentication is transferred when calling the API, the status code **401** is returned and the response result is as follows:

```
{"code":"1001","message":"incorrect username or password"}
```

Mapping Frontend Authentication Parameters to Backend Parameters

If the authentication is successful, the context information returned by the function can be transferred to the backend of the API. To do this, perform the following configurations:

On the **APIs** page, choose **More** > **Edit** in the row that contains the API, and go to the **Define Backend Request** page. Add a system parameter, specify the parameter type as **Frontend authentication parameter**, set the parameter name to the content of the **context** field in the function response, and set the name and location of the backend parameter to which you want to the map the frontend authentication parameter.



Configure VPC Channel	Configure now / Do not configure
	Specify a VPC channel to access services deployed in VPCs.
* VPC Channel	VPC_dcoi ▾ Manage VPC Channel
Host Header	
	The host header can be customized for requests to be forwarded to cloud servers through the VPC channel. By default, the original host header of the request is used.
* Path	/backend
	Enter a path and enclose the parameters in braces, for example, /getUserInfo/{userId}. The following special characters are allowed: *%+-_.
* Timeout (ms)	5000
Backend Authentication	
	Specify a custom authorizer to control access to the backend service.

You can create 49 more backend, constant, and system parameters (max. backend, constant, and system parameters: 50).

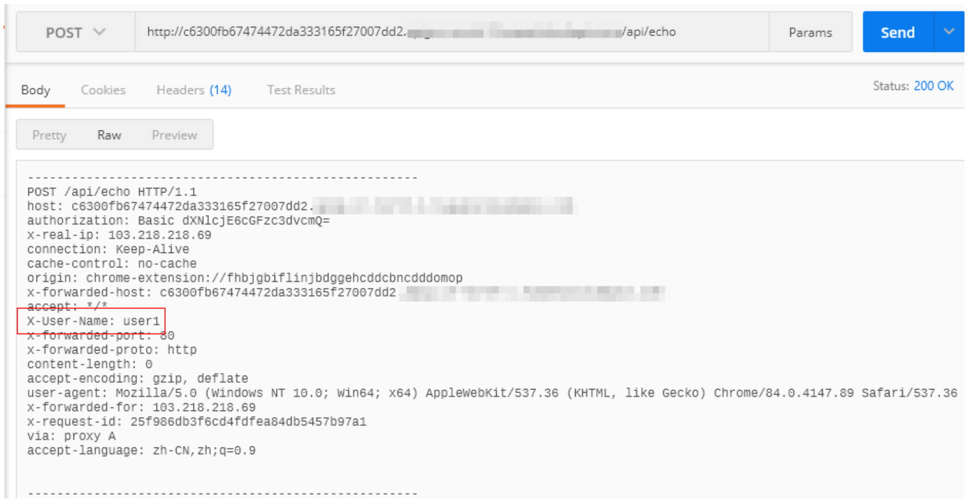
Backend Parameters

Constant Parameters

System Parameters

System Parameter Type	System Parameter Name	Backend Parameter Name	Backend Parameter Location	Description	Operation
Frontend authentication parameter ▾	user_name	X-User-Name	HEADER ▾	Enter a parameter description.	Delete

After modifying the API, publish it again. If the authentication information carried in a request for the API is correct, the response result contains the **X-User-Name** header field whose value is the same as that of **user_name** in the **context** field of the authentication function.



http://c6300fb67474472da333165f27007dd2.[redacted]/api/echo

Status: 200 OK

```
----------------------------------------------------
POST /api/echo HTTP/1.1
host: c6300fb67474472da333165f27007dd2.[redacted]
authorization: Basic dXNlcjE6cGFzc3dvcmQ=
x-real-ip: 103.218.218.69
connection: Keep-Alive
cache-control: no-cache
origin: chrome-extension://fhbjgbiflinjbdggehcddcbncdddomop
x-forwarded-host: c6300fb67474472da333165f27007dd2[redacted]
accept: */*
X-User-Name: user1
x-forwarded-port: 80
x-forwarded-proto: http
content-length: 0
accept-encoding: gzip, deflate
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.89 Safari/537.36
x-forwarded-for: 103.218.218.69
x-request-id: 25f986db3f6cd4fdfea84db5457b97a1
via: proxy A
accept-language: zh-CN,zh;q=0.9


----------------------------------------------------
```

4 Exposing Backend Services Across VPCs

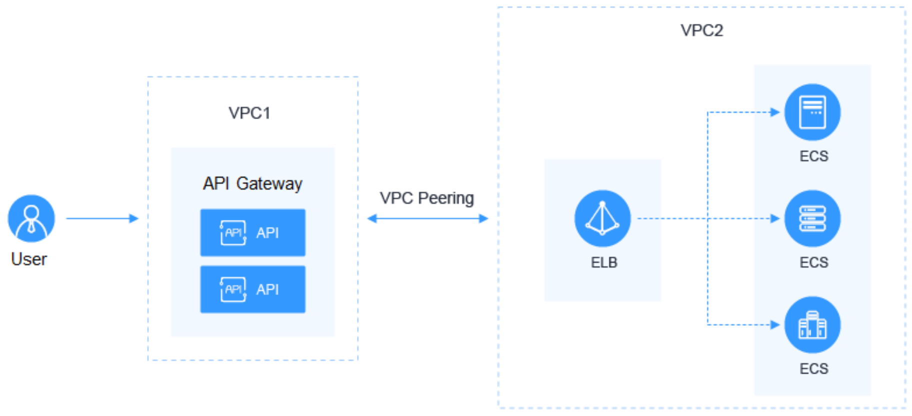
4.1 Introduction

Scenario

If the VPC of your backend server is different from that of your gateway, how do you configure cross-VPC interconnection? This section uses Elastic Load Balance (ELB) as an example to describe how to expose services in a private network load balancer using APIG.

Solution Architecture

Figure 4-1 Exposing backend services across VPCs



Advantages

Without modifying the existing network architecture, you can have all requests directly forwarded to your backend server through flexible configuration.

Restrictions

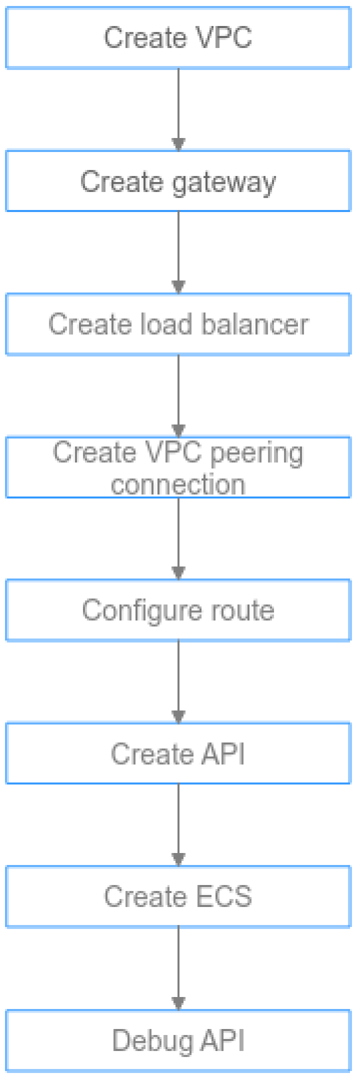
VPC 1, VPC 2, and the VPC CIDR block of your gateway cannot overlap. For details about the VPC CIDR block planning of the gateway, see **Table 4-3**.

4.2 Resource Planning

Table 4-1 Resource planning

Resource	Quantity
VPC	2
Dedicated gateway	1
Load balancer	1
ECS	1

4.3 General Procedure



1. **Create a VPC.**

   Create two VPCs, one for your gateway and the other for your backend service.
2. **Create a gateway.**

   Create a dedicated gateway in VPC 1.
3. **Create a load balancer.**

   Create a load balancer in VPC 2.
4. **Create a VPC peering connection.**

   Create a VPC peering connection to connect VPC 1 and VPC 2.
5. **Configure a route.**

   Configure a route for the dedicated gateway by setting the IP address to the IPv4 CIDR block of VPC 2 where the created load balancer is located.
6. **Create an API.**

   Create an API and set the backend service address to the IP address of the load balancer.
7. **Create an ECS.**

   Create an ECS in VPC 2, and deploy the backend service on the ECS.
8. **Debug the API.**

   Verify that the connection to the private network load balancer is successful.

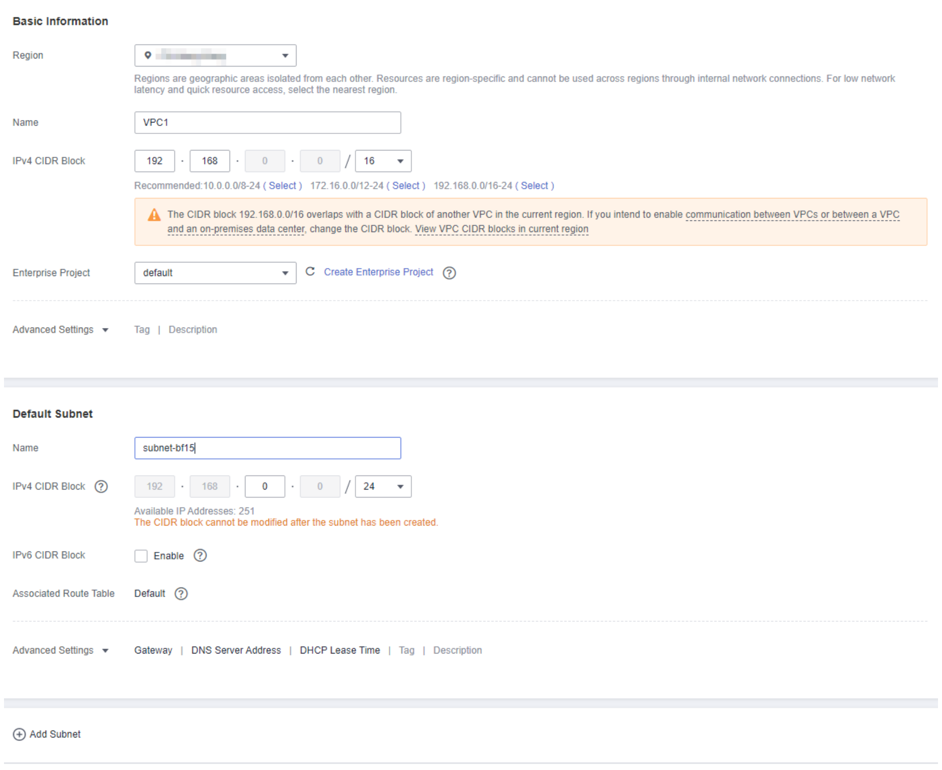
4.4 Implementation Procedure

Creating a VPC

Step 1 Log in to the network console.

Step 2 In the navigation pane, choose **Virtual Private Cloud** > **My VPCs**.

Step 3 On the **Virtual Private Cloud** page, click **Create VPC**, and configure the parameters by referring to **Table 4-2** and **Table 4-3**. For details, see sections "Creating a VPC" and "Creating a Subnet for the VPC" in the *Virtual Private Cloud User Guide*.



Basic Information

Region: [redacted]

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

Name: VPC1

IPv4 CIDR Block: 192 . 168 . 0 . 0 / 16

Recommended:10.0.0.0/8-24 (Select)   172.16.0.0/12-24 (Select)   192.168.0.0/16-24 (Select)

The CIDR block 192.168.0.0/16 overlaps with a CIDR block of another VPC in the current region. If you intend to enable communication between VPCs or between a VPC and an on-premises data center, change the CIDR block. View VPC CIDR blocks in current region

Enterprise Project: default

Advanced Settings: Tag | Description

Default Subnet

Name: subnet-bf15

IPv4 CIDR Block: 192 . 168 . 0 . 0 / 24

Available IP Addresses: 251
The CIDR block cannot be modified after the subnet has been created.

IPv6 CIDR Block: ☐ Enable

Associated Route Table: Default

Advanced Settings: Gateway | DNS Server Address | DHCP Lease Time | Tag | Description

Table 4-2 Configuration information

Parameter	Description
Region	Select a region.
Name	Enter **VPC1**. This VPC will be used to run a gateway.
Enterprise Project	Select **default**.
Name	A subnet is automatically created when you create a VPC.

Table 4-3 VPC CIDR block planning

VPC 1	VPC of APIG	VPC 2
10.X	172.31.0.0/16	Must be different from VPC 1 and the VPC of the gateway.
172.X	192.168.0.0/16	
192.X	172.31.0.0/16	

Step 4 Click **Create Now**.

Step 5 Repeat **Step 3** to **Step 4** to create **VPC2** for running your backend service.

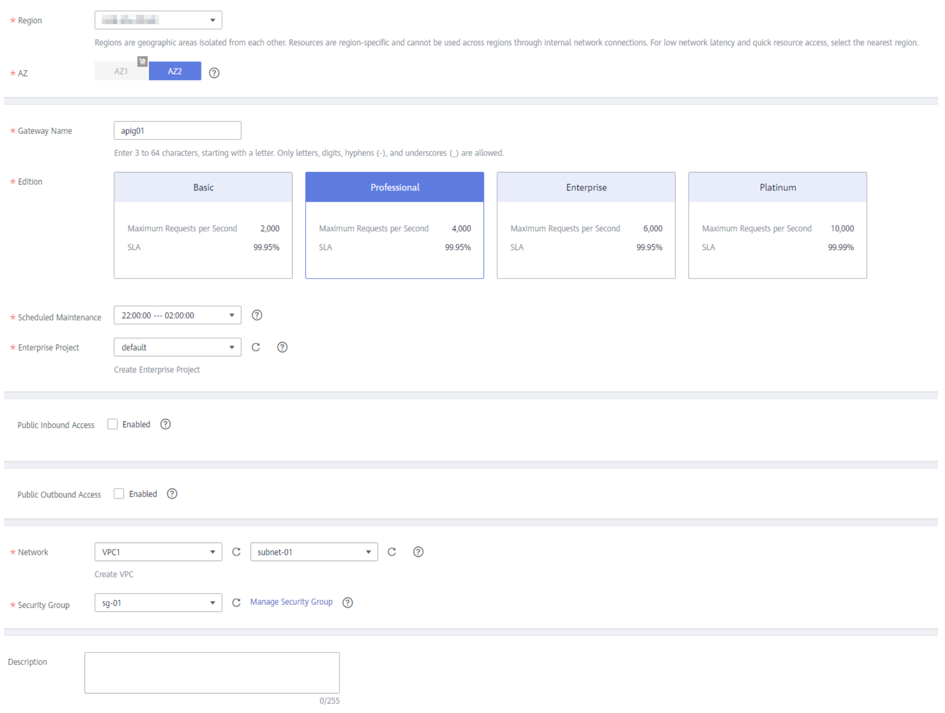
----End

Creating a Gateway

Step 1 Log in to the APIG console.

Step 2 In the navigation pane, choose **Gateways**.

Step 3 Click **Buy Gateway**.



* Region [redacted]

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

* AZ AZ1 [✓ AZ2]

* Gateway Name apig01

Enter 3 to 64 characters, starting with a letter. Only letters, digits, hyphens (-), and underscores (_) are allowed.

* Edition

Basic	[✓ Professional]	Enterprise	Platinum
Maximum Requests per Second 2,000	Maximum Requests per Second 4,000	Maximum Requests per Second 6,000	Maximum Requests per Second 10,000
SLA 99.95%	SLA 99.95%	SLA 99.95%	SLA 99.99%

* Scheduled Maintenance 22:00:00 --- 02:00:00

* Enterprise Project default

Create Enterprise Project

Public Inbound Access ☐ Enabled

Public Outbound Access ☐ Enabled

* Network VPC1 subnet-01

Create VPC

* Security Group sg-01 Manage Security Group

Description

0/255

Table 4-4 Gateway information

Parameter	Description
Region	Select the region where the gateway is located. It must be the same as the region of VPC 1.
AZ	The AZ where the gateway is located. Select **AZ1**.
Gateway Name	Enter a name that conforms to specific rules to facilitate search.
Edition	Select **Professional**. The edition cannot be changed after the gateway is created.
Scheduled Maintenance	Select a time period when the gateway can be maintained by technical support engineers. A period with low service traffic is recommended. For this example, retain the default value **22:00:00---02:00:00**.
Enterprise Project	Select the enterprise project to which the gateway belongs. For this example, retain the default value **default**.
Network	Select **VPC 1** and a subnet.

Parameter	Description
Security Group	Click **Manage Security Groups** and create a security group. Ensure that you have selected **default** for **Enterprise Project**.
Description	Description of the gateway.

Step 4 Click **Next**.

Step 5 If the gateway configurations are correct, and click **Pay Now**.

----End

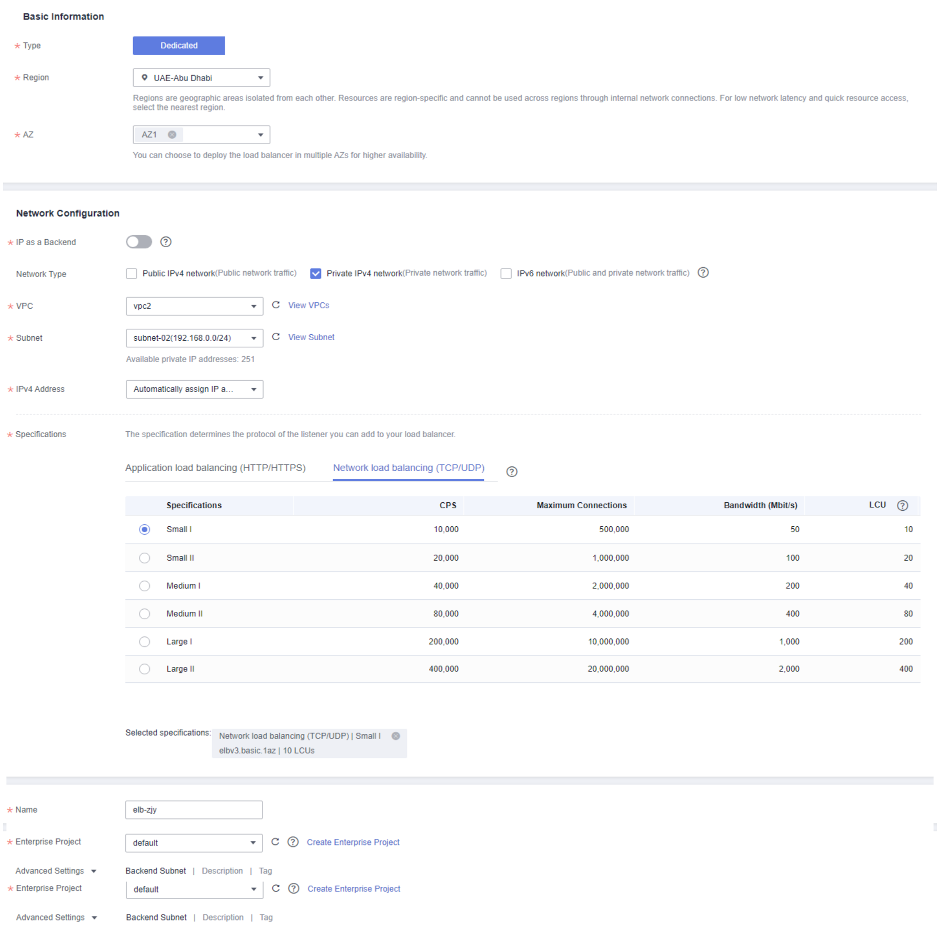
Creating a Load Balancer

Step 1 Log in to the network console.

Step 2 In the navigation pane, choose **Elastic Load Balance** > **Load Balancers**.

Step 3 Click **Create Elastic Load Balancer**.

Step 4 Configure the load balancer information. For details, see section **Load Balancer** in the *Elastic Load Balance User Guide*.



Basic Information

* Type	Dedicated
* Region	UAE-Abu Dhabi
	Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.
* AZ	AZ1
	You can choose to deploy the load balancer in multiple AZs for higher availability.

Network Configuration

* IP as a Backend	
Network Type	☐ Public IPv4 network(Public network traffic) ✓ Private IPv4 network(Private network traffic) ☐ IPv6 network(Public and private network traffic)
* VPC	vpc2
* Subnet	subnet-02(192.168.0.0/24)
	Available private IP addresses: 251
* IPv4 Address	Automatically assign IP a...
* Specifications	The specification determines the protocol of the listener you can add to your load balancer.

Application load balancing (HTTP/HTTPS) | Network load balancing (TCP/UDP)

	Specifications	CPS	Maximum Connections	Bandwidth (Mbit/s)	LCU
●	Small I	10,000	500,000	50	10
○	Small II	20,000	1,000,000	100	20
○	Medium I	40,000	2,000,000	200	40
○	Medium II	80,000	4,000,000	400	80
○	Large I	200,000	10,000,000	1,000	200
○	Large II	400,000	20,000,000	2,000	400

Selected specifications: Network load balancing (TCP/UDP) | Small I
elbv3.basic.1az | 10 LCUs

* Name	elb-zjy
* Enterprise Project	default
Advanced Settings	Backend Subnet | Description | Tag
* Enterprise Project	default
Advanced Settings	Backend Subnet | Description | Tag

Table 4-5 Load balancer parameters

Parameter	Description
Type	Type of the load balancer.
Region	Select the region where the load balancer is located. It must be the same as the region of VPC 2.
AZ	The AZ where the load balancer is located. Select **AZ1**.
Network Type	Select **Private Network**.
VPC	Select **VPC 2**.
Subnet	Select a subnet.
Specification	Select **Network load balancing**.
Name	Enter a load balancer name that conforms to specific rules to facilitate search.
Enterprise Project	Select **default**.

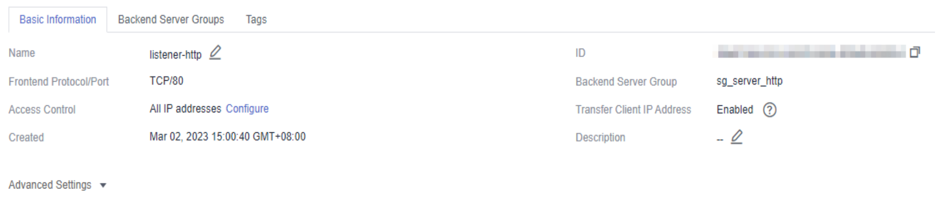
Step 5 Click **Create Now**.

Step 6 Confirm the configuration and click **Submit**.

Step 7 Add a listener.

1. Click the name of the load balancer. On the **Listeners** tab page, click **Add Listener**.
2. Configure the listener name, frontend protocol, and port, and click **Next**.
3. Configure the backend server group name, backend protocol, and load balancing algorithm. Then click **Next**.
4. Add backend servers and click **Next**.
5. Click **Submit**. The following figure shows the configuration.

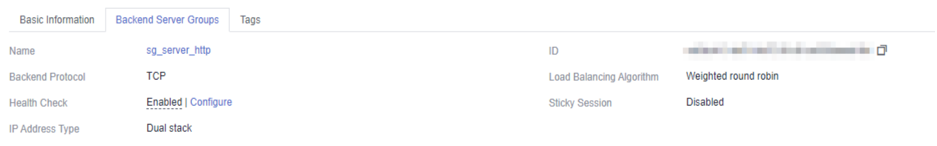
Figure 4-2 Listener information



Name	listener-http	ID	[redacted]
Frontend Protocol/Port	TCP/80	Backend Server Group	sg_server_http
Access Control	All IP addresses Configure	Transfer Client IP Address	Enabled
Created	Mar 02, 2023 15:00:40 GMT+08:00	Description	--

Advanced Settings

Figure 4-3 Backend server group information



Name	sg_server_http	ID	[redacted]
Backend Protocol	TCP	Load Balancing Algorithm	Weighted round robin
Health Check	Enabled | Configure	Sticky Session	Disabled
IP Address Type	Dual stack		

----End

Creating a VPC Peering Connection

Step 1 Log in to the network console.

Step 2 In the navigation pane, choose **Virtual Private Cloud** > **VPC Peering Connections**.

Step 3 Click **Create VPC Peering Connection** and configure the parameters.

Table 4-6 Configuring a VPC peering connection

Parameter	Description
Name	Enter a VPC peering connection name that conforms to specific rules to facilitate search.
Local VPC	Select **VPC 1**.
Account	By default, **My account** is selected.
Peer Project	Select a project
Peer VPC	Select **VPC 2**.

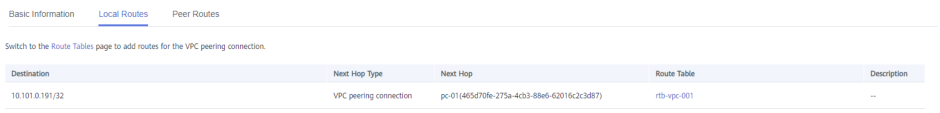
Step 4 Click **OK**.

Step 5 In the displayed dialog box, click **Add Route** to go to the VPC peering connection details page.

Step 6 On the **Local Routes** tab page, click **Route Tables**.

1. Under **Routes**, click **Add Route**.
2. In the displayed dialog box, enter the route information.
   - **Destination**: Enter the service address displayed on the details page of the **load balancer**.
   - **Next Hop Type**: Select **VPC peering connection**.
3. Click **OK**.

Figure 4-4 Local routes



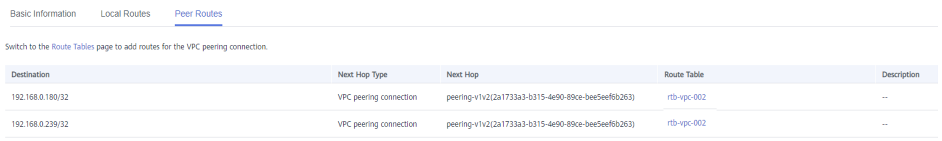
Switch to the Route Tables page to add routes for the VPC peering connection.

Destination	Next Hop Type	Next Hop	Route Table	Description
10.101.0.191/32	VPC peering connection	pc-01(465d70fe-275a-4cb3-88e6-62016c2c3d87)	rtb-vpc-001	--

Step 7 Go to the **Peer Routes** tab page, and click **Route Tables**.

1. Under **Routes**, click **Add Route**.
2. In the displayed dialog box, enter the route information.
   - **Destination**: Enter the private outbound address displayed on the details page of the **dedicated gateway**.
   - **Next Hop Type**: Select **VPC peering connection**.
3. Click **OK**.

Figure 4-5 Peer routes



Switch to the Route Tables page to add routes for the VPC peering connection.

Destination	Next Hop Type	Next Hop	Route Table	Description
192.168.0.180/32	VPC peering connection	peering-v1v2(2a1733a3-b315-4e90-89ce-bee5eef6b263)	rtb-vpc-002	--
192.168.0.239/32	VPC peering connection	peering-v1v2(2a1733a3-b315-4e90-89ce-bee5eef6b263)	rtb-vpc-002	--

----End

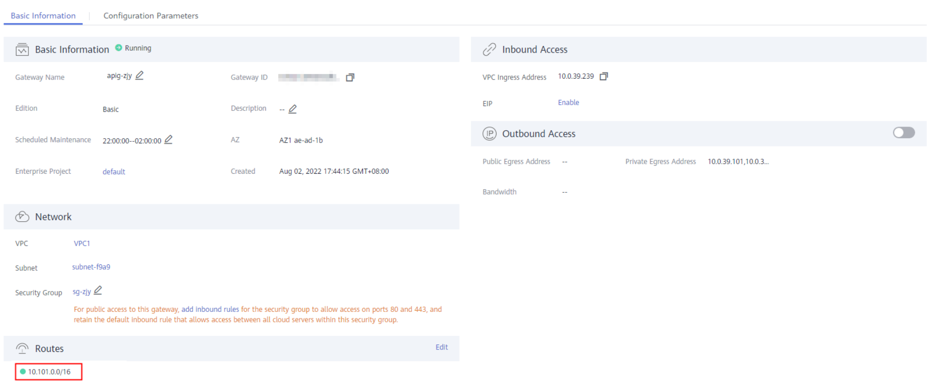
Configuring a Route

Step 1 Log in to the APIG console.

Step 2 In the navigation pane, choose **Gateways**.

Step 3 Click the name of the created **dedicated gateway** or click **Access Console**.

Step 4 Click **Change** in the **Routes** area, enter the IPv4 CIDR block of VPC 2 where the load balancer you created is located.



Basic Information Running

Gateway Name	apig-zjy	Gateway ID	[redacted]
Edition	Basic	Description	--
Scheduled Maintenance	22:00:00--02:00:00	AZ	AZ1 ae-ad-1b
Enterprise Project	default	Created	Aug 02, 2022 17:44:15 GMT+08:00

Network

VPC	VPC1
Subnet	subnet-f9a9
Security Group	sg-zjy

For public access to this gateway, add inbound rules for the security group to allow access on ports 80 and 443, and retain the default inbound rule that allows access between all cloud servers within this security group.

Routes Edit

● 10.101.0.0/16

Inbound Access

VPC Ingress Address	10.0.39.239
EIP	Enable

Outbound Access

Public Egress Address	--	Private Egress Address	10.0.39.101,10.0.3...
Bandwidth	--		

Step 5 Click **Save**.

----End

Creating an API

Step 1 Log in to the APIG console.

Step 2 In the upper part of the navigation pane, select the gateway.

Step 3 Choose **API Management** > **APIs**, and click **Create API**.

Step 4 Configure the frontend information and click **Next**.

Table 4-7 Frontend configuration

Parameter	Description
API Name	Enter a name that conforms to specific rules to facilitate search.
Group	The default option is **DEFAULT**.

Parameter	Description
URL	**Method**: Request method of the API. Set this parameter to **GET**. **Protocol**: Request protocol of the API. Set this parameter to **HTTPS**. **Subdomain Name**: The system automatically allocates a subdomain name to each API group for internal testing. The subdomain name can be accessed 1000 times a day. **Path**: Path for requesting the API.
Gateway Response	Select a response to be displayed if the gateway fails to process an API request. The default gateway response is **default**.
Authentication Mode	API authentication mode. Select **None**.

Step 5 Configure the backend information and click **Next**.

Table 4-8 Parameters for defining an HTTP/HTTPS backend service

Parameter	Description
Load Balance Channel	Determine whether the backend service will be accessed using a load balance channel. For this example, select **Skip**.
URL	**Method**: Request method of the API. Set this parameter to **GET**. **Protocol**: Set this parameter to **HTTP**. **Backend Address**: Enter the service address of the load balancer you created. **Path**: Path of the backend service.

Step 6 Define the response and click **Finish**.

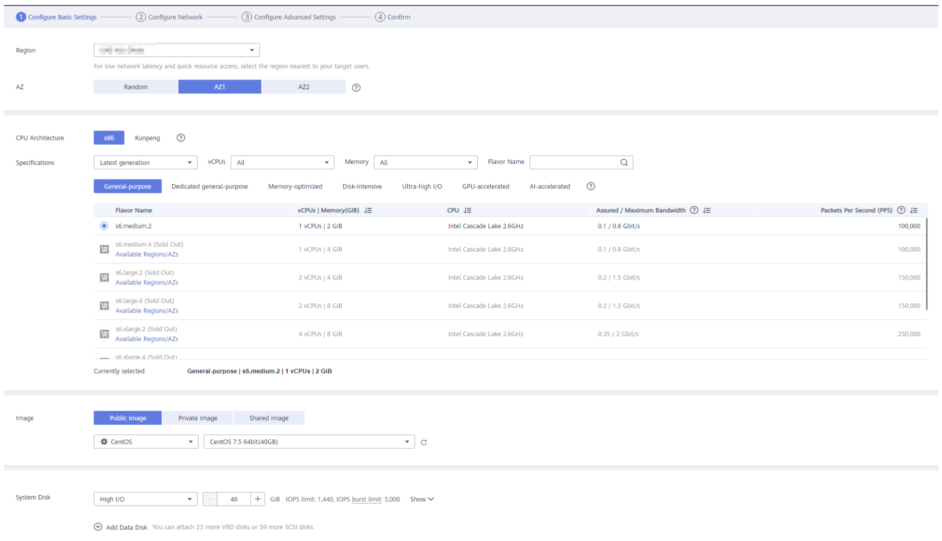
----End

Creating an ECS

Step 1 Log in to the cloud server console.

Step 2 Click **Create ECS**.

Step 3 Configure the basic settings and click **Next: Configure Network**.



① Configure Basic Settings — ② Configure Network — ③ Configure Advanced Settings — ④ Confirm

Region: [redacted]

For low network latency and quick resource access, select the region nearest to your target users.

AZ: Random [✓ AZ1] AZ2

CPU Architecture: [✓ x86] Kunpeng

Specifications: Latest generation | vCPUs: All | Memory: All | Flavor Name:

[✓ General-purpose] Dedicated general-purpose | Memory-optimized | Disk-intensive | Ultra-high I/O | GPU-accelerated | AI-accelerated

	Flavor Name	vCPUs | Memory(GiB)	CPU	Assured / Maximum Bandwidth	Packets Per Second (PPS)
●	s6.medium.2	1 vCPUs | 2 GiB	Intel Cascade Lake 2.6GHz	0.1 / 0.8 Gbit/s	100,000
	s6.medium.4 (Sold Out) Available Regions/AZs	1 vCPUs | 4 GiB	Intel Cascade Lake 2.6GHz	0.1 / 0.8 Gbit/s	100,000
	s6.large.2 (Sold Out) Available Regions/AZs	2 vCPUs | 4 GiB	Intel Cascade Lake 2.6GHz	0.2 / 1.5 Gbit/s	150,000
	s6.large.4 (Sold Out) Available Regions/AZs	2 vCPUs | 8 GiB	Intel Cascade Lake 2.6GHz	0.2 / 1.5 Gbit/s	150,000
	s6.xlarge.2 (Sold Out) Available Regions/AZs	4 vCPUs | 8 GiB	Intel Cascade Lake 2.6GHz	0.35 / 2 Gbit/s	250,000
	s6.xlarge.4 (Sold Out)				

Currently selected: **General-purpose | s6.medium.2 | 1 vCPUs | 2 GiB**

Image: [✓ Public Image] Private image | Shared image

CentOS | CentOS 7.5 64bit(40GB)

System Disk: High I/O | 40 GiB IOPS limit: 1,440, IOPS burst limit: 5,000 Show

Add Data Disk You can attach 23 more VBD disks or 59 more SCSI disks.

Table 4-9 Basic settings

Parameter	Description
Region	Select the region where the ECS is located. It must be the same as the region of VPC 2.
AZ	Select the AZ where the ECS is located.
CPU Architecture	The default option is **x86**.
Specifications	Select specifications that match your service planning.
Image	Select an image that matches your service planning.

Step 4 Configure the network settings and click **Next: Configure Advanced Settings**.

Table 4-10 Network settings

Parameter	Description
Network	Select **VPC 2** and a subnet.
Security Group	Select the security group created for the **dedicated gateway**.
EIP	Select **Not required**.

Step 5 Configure advanced settings and click **Next: Confirm**.

Table 4-11 Advanced settings

Parameter	Description
ECS Name	Enter a name that conforms to specific rules to facilitate search.
Login Mode	Credential for logging in to the ECS. The default option is **Password**.
Username	The default user is **root**.
Password	Set a password for logging in to the ECS.
Confirm Password	Enter the password again.

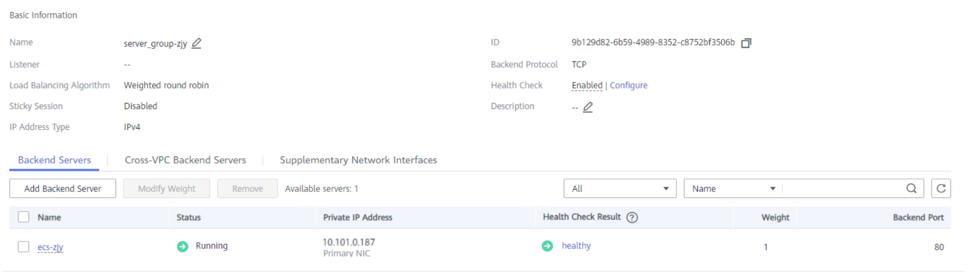
Step 6 Confirm the configuration and select enterprise project **default**.

Step 7 Read and confirm your acceptance of the agreement. Then click **Create Now**.

----End

Debugging the API

Step 1 On the **Backend Server Groups** tab page of **the load balancer**, add **the ECS**.


Basic Information

Name	server_group-zjy	ID	9b129d82-6b59-4989-8352-c8752bf3506b
Listener	--	Backend Protocol	TCP
Load Balancing Algorithm	Weighted round robin	Health Check	Enabled | Configure
Sticky Session	Disabled	Description	--
IP Address Type	IPv4		

Available servers: 1

Name	Status	Private IP Address	Health Check Result	Weight	Backend Port
ecs-zjy	Running	10.101.0.187 Primary NIC	healthy	1	80

Step 2 Start the ECS.

Step 3 Go to the **API Management** > **APIs** page of **the dedicated gateway**, and choose **More** > **Debug** in the row that contains **the API you created**.

Step 4 Enter the request parameters and click **Debug**.

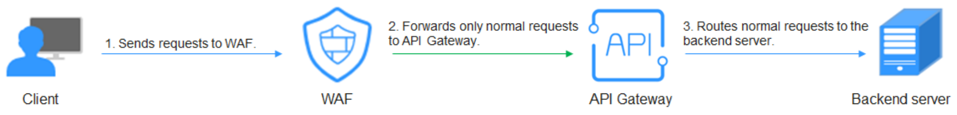
If the status code is **200**, the debugging is successful.

----End

5 Interconnecting with WAF

To protect API Gateway and your backend servers from malicious attacks, deploy Web Application Firewall (WAF) between API Gateway and the external network.

Figure 5-1 Access to a backend server

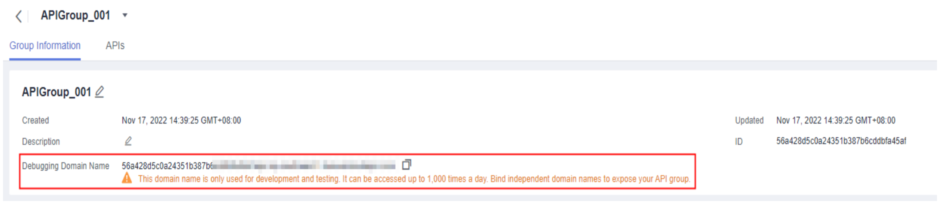

Client
1. Sends requests to WAF.
WAF
2. Forwards only normal requests to API Gateway.
API
API Gateway
3. Routes normal requests to the backend server.
Backend server

(Recommended) Solution 1: Register API Group Debugging Domain Name on WAF and Use the Domain Name to Access the Backend Service

API groups provide services using domain names for high scalability.

Step 1 Create an API group in a gateway, record the domain name, and create an API in the group.

Figure 5-2 Creating an API group and recording the debugging domain name

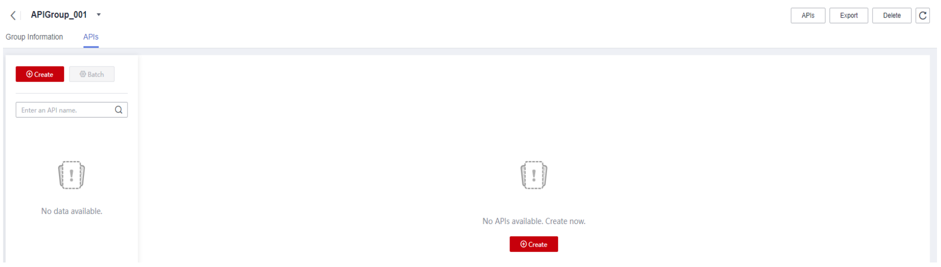


APIGroup_001

Created	Nov 17, 2022 14:39:25 GMT+08:00	Updated	Nov 17, 2022 14:39:25 GMT+08:00
Description		ID	56a428d5c0a24351b387b6cddbfa45af
Debugging Domain Name	56a428d5c0a24351b387b6[redacted]		

This domain name is only used for development and testing. It can be accessed up to 1,000 times a day. Bind independent domain names to expose your API group.

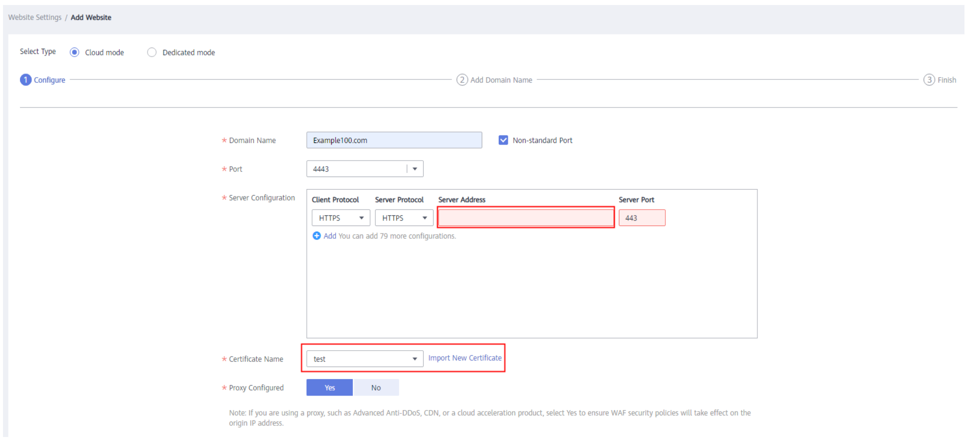
Figure 5-3 Creating an API



Step 2 Go to the WAF console, and add a domain name by configuring **Server Address** as the API group domain name and adding a certificate. For details, see section "Connection Process (Cloud Mode)" in the *Web Application Firewall User Guide*.

NOTE

You can use a public network client to access WAF with its domain name. WAF then uses the same domain name to forward your requests to API Gateway. There is no limit on the number of requests that API Gateway can receive for the domain name.


Website Settings / Add Website

Select Type: [✓ Cloud mode] Dedicated mode

1 Configure — 2 Add Domain Name — 3 Finish

* Domain Name: Example100.com ✓ Non-standard Port

* Port: 4443

* Server Configuration:

Client Protocol	Server Protocol	Server Address	Server Port
HTTPS	HTTPS		443

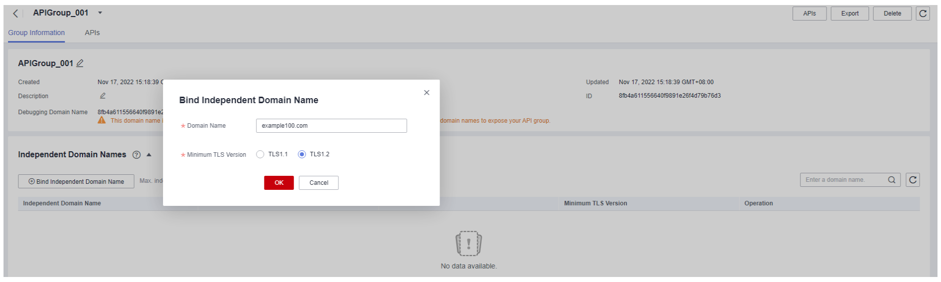
Add You can add 79 more configurations.

* Certificate Name: test Import New Certificate

* Proxy Configured: [✓ Yes] No

Note: If you are using a proxy, such as Advanced Anti-DDoS, CDN, or a cloud acceleration product, select Yes to ensure WAF security policies will take effect on the origin IP address.

Step 3 On the gateway details page, bind the domain name to the API group.


APIGroup_001

Group Information APIs

APIGroup_001

Created: Nov 17, 2022 15:18:39 G
Description:
Debugging Domain Name: 8fb4a611556640f9891e2
This domain name … domain names to expose your API group.

Updated: Nov 17, 2022 15:18:39 GMT+08:00
ID: 8fb4a611556640f9891e26f4d79b76d3

Independent Domain Names

Bind Independent Domain Name Max. ind

Independent Domain Name	Minimum TLS Version	Operation

No data available.

Bind Independent Domain Name

* Domain Name: example100.com

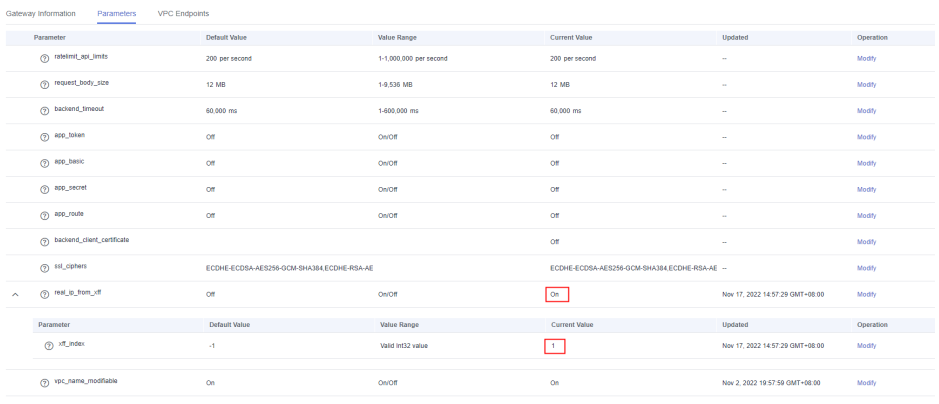
* Minimum TLS Version: TLS1.1 [✓ TLS1.2]

OK Cancel

Step 4 Enable **real_ip_from_xff** and set the parameter value to **1**.

NOTE

When a user accesses WAF using a public network client, WAF records the actual IP address of the user in the HTTP header **X-Forwarded-For**. API Gateway resolves the actual IP address of the user based on the header.



Parameter	Default Value	Value Range	Current Value	Updated	Operation
ratelimit_api_limits	200 per second	1-1,000,000 per second	200 per second	--	Modify
request_body_size	12 MB	1-9,536 MB	12 MB	--	Modify
backend_timeout	60,000 ms	1-600,000 ms	60,000 ms	--	Modify
app_token	Off	On/Off	Off	--	Modify
app_basic	Off	On/Off	Off	--	Modify
app_secret	Off	On/Off	Off	--	Modify
app_route	Off	On/Off	Off	--	Modify
backend_client_certificate			Off	--	Modify
ssl_ciphers	ECDHE-ECDSA-AES256-GCM-SHA384,ECDHE-RSA-AE		ECDHE-ECDSA-AES256-GCM-SHA384,ECDHE-RSA-AE	--	Modify
real_ip_from_xff	Off	On/Off	On	Nov 17, 2022 14:57:29 GMT+08:00	Modify

Parameter	Default Value	Value Range	Current Value	Updated	Operation
xff_index	-1	Valid Int32 value	1	Nov 17, 2022 14:57:29 GMT+08:00	Modify

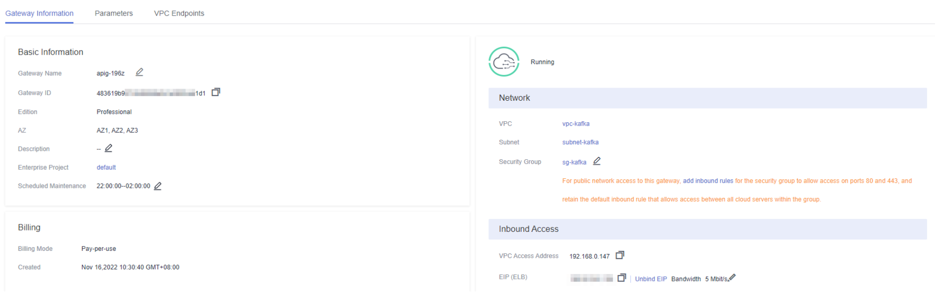
Parameter	Default Value	Value Range	Current Value	Updated	Operation
vpc_name_modifiable	On	On/Off	On	Nov 2, 2022 19:57:59 GMT+08:00	Modify

----End

Solution 2: Forward Requests Through the DEFAULT Group and Use Gateway Inbound Access Address to Access the Backend Service from WAF

Step 1 View the inbound access addresses of your gateway. There is no limit on the number of times the API gateway can be accessed using an IP address.

- **VPC Ingress Address**: VPC access address
- **EIP**: public network access address



Basic Information

Gateway Name	apig-196z
Gateway ID	483619b9[redacted]1d1
Edition	Professional
AZ	AZ1, AZ2, AZ3
Description	--
Enterprise Project	default
Scheduled Maintenance	22:00:00--02:00:00

Billing

Billing Mode	Pay-per-use
Created	Nov 16,2022 10:30:40 GMT+08:00

Network

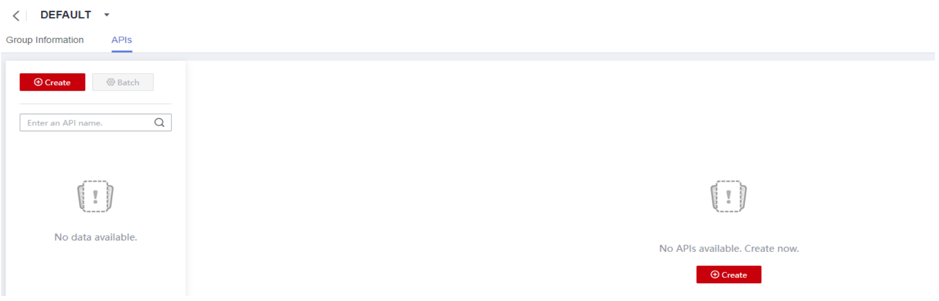
VPC	vpc-kafka
Subnet	subnet-kafka
Security Group	sg-kafka

For public network access to this gateway, add inbound rules for the security group to allow access on ports 80 and 443, and retain the default inbound rule that allows access between all cloud servers within the group.

Inbound Access

VPC Access Address	192.168.0.147
EIP (ELB)	[redacted] | Unbind EIP Bandwidth 5 Mbit/s

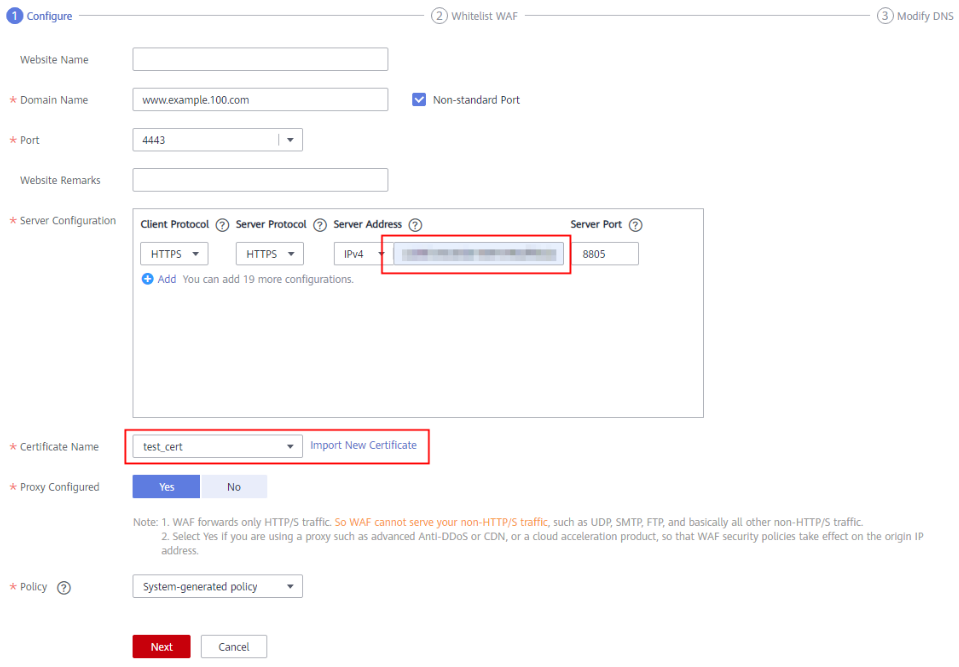
Step 2 Create an API in the **DEFAULT** group.



Step 3 Go to the WAF console, add a domain name by configuring **Server Address** as an **inbound access address** of your API gateway and adding a certificate, and then copy the WAF back-to-source IP addresses. For details, see section "Connection Process (Cloud Mode)" in the *Web Application Firewall User Guide*.

NOTE

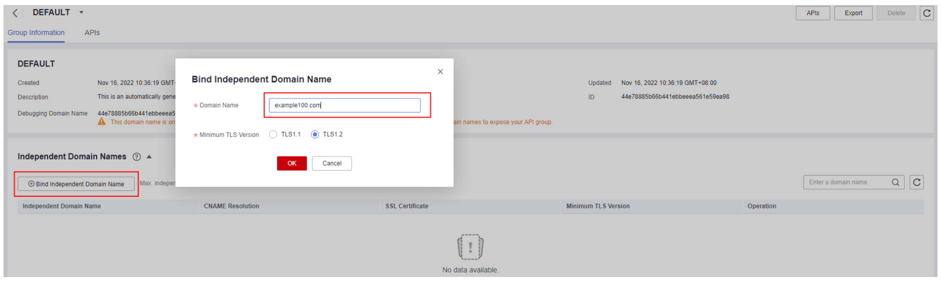
- If WAF and your gateway are in the same VPC, set **Server Address** to the VPC access address.
- If your gateway is bound with an EIP, set **Server Address** to the EIP.



Website Name	
* Domain Name	www.example.100.com [✓ Non-standard Port]
* Port	4443
Website Remarks	
* Server Configuration	Client Protocol: HTTPS; Server Protocol: HTTPS; Server Address: IPv4 [redacted]; Server Port: 8805
	Add You can add 19 more configurations.
* Certificate Name	test_cert Import New Certificate
* Proxy Configured	[✓ Yes] No
* Policy	System-generated policy

Note: 1. WAF forwards only HTTP/S traffic. So WAF cannot serve your non-HTTP/S traffic, such as UDP, SMTP, FTP, and basically all other non-HTTP/S traffic.
2. Select Yes if you are using a proxy such as advanced Anti-DDoS or CDN, or a cloud acceleration product, so that WAF security policies take effect on the origin IP address.

Step 4 On the gateway details page, bind the domain name to the **DEFAULT** group.



DEFAULT

Created	Nov 16, 2022 10:36:19 GMT-	Updated	Nov 16, 2022 10:36:19 GMT+08:00
Description	This is an automatically gene	ID	44e78885b66b441ebbeeea561e59ea98
Debugging Domain Name	44e78885b66b441ebbeeea5		

This domain name is on ... ain names to expose your API group.

Bind Independent Domain Name
* Domain Name: example100.com
* Minimum TLS Version: TLS1.1 [✓ TLS1.2]

Independent Domain Names
Max. indepen

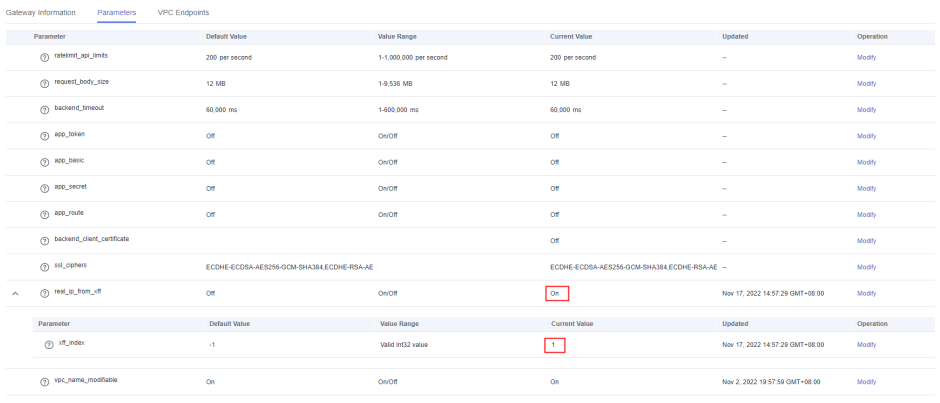
Independent Domain Name	CNAME Resolution	SSL Certificate	Minimum TLS Version	Operation

No data available.

Step 5 Enable **real_ip_from_xff** and set the parameter value to **1**.

NOTE

When a user accesses WAF using a public network client, WAF records the actual IP address of the user in the HTTP header **X-Forwarded-For**. API Gateway resolves the actual IP address of the user based on the header.


Gateway Information | Parameters | VPC Endpoints

Parameter	Default Value	Value Range	Current Value	Updated	Operation
ratelimit_api_limits	200 per second	1-1,000,000 per second	200 per second	--	Modify
request_body_size	12 MB	1-9,536 MB	12 MB	--	Modify
backend_timeout	60,000 ms	1-600,000 ms	60,000 ms	--	Modify
app_token	Off	On/Off	Off	--	Modify
app_basic	Off	On/Off	Off	--	Modify
app_secret	Off	On/Off	Off	--	Modify
app_route	Off	On/Off	Off	--	Modify
backend_client_certificate			Off	--	Modify
ssl_ciphers	ECDHE-ECDSA-AES256-GCM-SHA384,ECDHE-RSA-AE		ECDHE-ECDSA-AES256-GCM-SHA384,ECDHE-RSA-AE	--	Modify
real_ip_from_xff	Off	On/Off	On	Nov 17, 2022 14:57:29 GMT+08:00	Modify
vpc_name_modifiable	On	On/Off	On	Nov 2, 2022 19:57:59 GMT+08:00	Modify

Parameter	Default Value	Value Range	Current Value	Updated	Operation
xff_index	-1	Valid Int32 value	1	Nov 17, 2022 14:57:29 GMT+08:00	Modify

----End

6 Request Throttling 2.0

6.1 Introduction

Scenario

If the number of requests initiated from public networks for open APIs on APIG is not limited, the continuous increase in users will deteriorate the backend performance. And what's worse, the website or program will break down due to a large number of requests sent by malicious users. The traditional request throttling policies of APIG throttle requests by API, user, credential, and source IP address.

However, as users and their demands become more diversified, these traditional policies cannot meet the requirements for more refined rate limiting. To resolve this issue, APIG has launched request throttling 2.0, which is a type of plug-in policy. The 2.0 policies enable you to configure more refined throttling, for example, to throttle requests based on a certain request parameter or tenant.

This section describes how to create a request throttling 2.0 policy for rate limiting in different scenarios.

Advantages

- A request throttling 2.0 policy limits the number of times that an API can be called within a specific time period. Basic, parameter-based, and excluded throttling is supported.
  - Basic throttling: Throttle requests by API, user, credential, or source IP address. This function is similar to a traditional request throttling policy but is incompatible with it.
  - Parameter-based throttling: Throttle requests based on headers, path parameter, method, query strings, or system parameters.
  - Excluded throttling: Throttle requests for specific tenants or credentials.
- API requests allowed in a time period can be limited by user or credential.
- Request throttling can be precise to the day, hour, minute, or second.

Restrictions

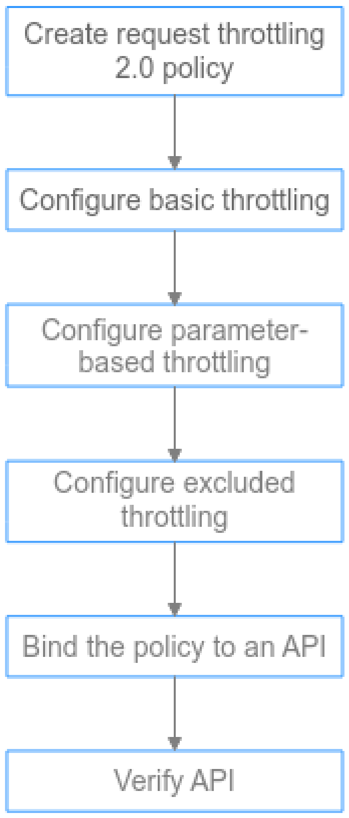
- Adding a request throttling 2.0 policy to an API means binding them together. An API can be bound with only one such policy in an environment, but each policy can be bound to multiple APIs. The APIs bound with request throttling 2.0 policies must have been published.
- For APIs not bound with a request throttling 2.0 policy, the throttling limit is the value of **ratelimit_api_limits** set on the **Parameters** page of the gateway.
- A traditional request throttling policy becomes invalid if a request throttling 2.0 policy is bound to the same API as the traditional one.
- You can define a maximum of 100 parameter-based throttling rules.
- The policy content cannot exceed 65,535 characters.
- If your gateway does not support request throttling 2.0, contact technical support.

6.2 General Procedure

Assume that you have the following request throttling requirements for an API:

1. The API can be called up to 10 times per 60s but can be called by a user only 5 times per 60s.
2. Only 10 requests containing header field **Host=www.abc.com** are allowed in 60s.
3. Only 10 requests with method **GET** and path **reqPath = /list** are allowed in 60s.
4. Only 10 requests with path **reqPath = /fc** are allowed in 60s.
5. Each excluded tenant can only call the API 5 times per 60s.

Following this procedure to create a request throttling 2.0 policy and bind it to an API.


Create request throttling 2.0 policy
Configure basic throttling
Configure parameter-based throttling
Configure excluded throttling
Bind the policy to an API
Verify API

1. **Create a policy.**
   Enter the basic information of the request throttling 2.0 policy.
2. **Configure basic throttling.**
   Configure the basic throttling settings.
3. **Configure parameter-based throttling.**
   Enable parameter-based throttling, and define parameters and rules.
4. **Configure excluded throttling.**
   Enable excluded throttling, and configure excluded tenants and credentials.
5. **Bind the policy to an API.**
   Bind the request throttling 2.0 policy to the API.
6. **Verify the API.**
   Call the API and verify whether the request throttling 2.0 policy has taken effect.

6.3 Implementation Procedure

Step 1 Create a policy.

Log in to the APIG console and create a request throttling 2.0 policy. For details, see section "Request Throttling 2.0" in the *API Gateway User Guide*.

In the navigation pane, choose **API Management** > **API Policies**. Click **Create Policy**, and select **Request Throttling 2.0**.

Configure basic policy information to meet your demands.

Table 6-1 Policy basic Information

Parameter	Description
Name	Enter a policy name that conforms to specific rules to facilitate search.
Throttling	Select **High-performance**.
Policy Type	Select **API-specific**, which means measuring and throttling requests of a single API.
Period	Throttling period. Set this parameter to 60s.

Step 2 Configure basic throttling.

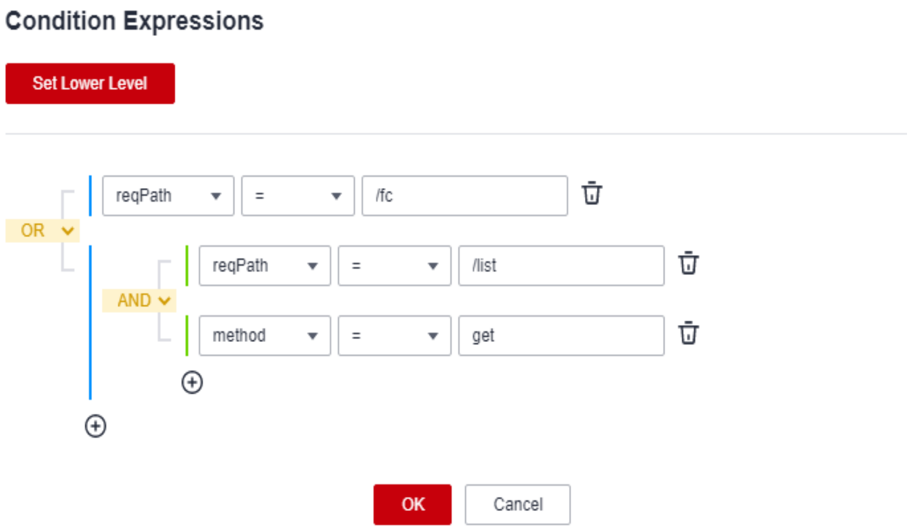
As required in 1, set **Max. API Requests** to 10 times per 60s and **Max. User Requests** to 5 times per 60s.

Table 6-2 Basic throttling

Parameter	Description
Max. API Requests	10
Max. User Requests	5

Step 3 Configure parameter-based throttling.

1. As required in 2, enable parameter-based throttling, and define the header and rule.
   a. Click **Add Parameter**, select **header** for **Parameter Location**, and enter **Host** for **Parameter**.
   b. In the **Rules** area, click **Add Rule**, and set **Max. API Requests** to **10** and **Period** to **60 seconds**. Then click ✎ , and set the matching condition **Host = www.abc.com**.
   c. Click **OK**. The header matching rule **Host = www.abc.com** is generated, indicating that an API bound with this policy can only be called 10 times per 60s by requests whose **Host** header is **www.abc.com**.
2. As required in 3 and 4, define multiple rules with parameter **Path**.
   a. In the **Rules** area, click **Add Rule**, and set **Max. API Requests** to **10** and **Period** to **60 seconds**. Then click ✎ to open the **Condition Expressions** dialog box.
   b. Add these three condition expressions: **reqPath = /fc**, **reqPath = /list**, and **method = get**.
   c. Click **Set Lower Level**.
   d. Put the two **reqPath** expressions in an **OR** relationship. This means the condition is met when either of the two paths is matched.
   e. Select **reqPath = /list** and **method = get**, click **Set Lower Level**, and select **AND**.



f. Click **OK**. It indicates that APIs with path **/list** and method **GET** or APIs with path **/fc** bound with this policy can only be called 10 times per 60s.

Step 4 Configure excluded throttling.

As required in 5, enable excluded throttling. Add an excluded tenant with a threshold of 5 requests per 60s.

Table 6-3 Excluded throttling

Parameter	Description
Account ID	Tenant ID
Threshold	5

Step 5 Click **OK**. The request throttling 2.0 policy is configured.

Step 6 Bind this policy to an API.

1. Click the policy name to go to the policy details page.
2. In the **APIs** area, select environment **RELEASE** and click **Bind to APIs**. Select an API and click **OK**.

Step 7 Verify the API.

Call the API and verify whether the request throttling 2.0 policy has taken effect.

----End

7 Two-Factor Authentication

7.1 Introduction

Scenario

APIG provides flexible authentication modes and allows you to configure a custom authorizer for two-factor authentication. This section describes how to create an API that uses two-factor authentication (app + custom).

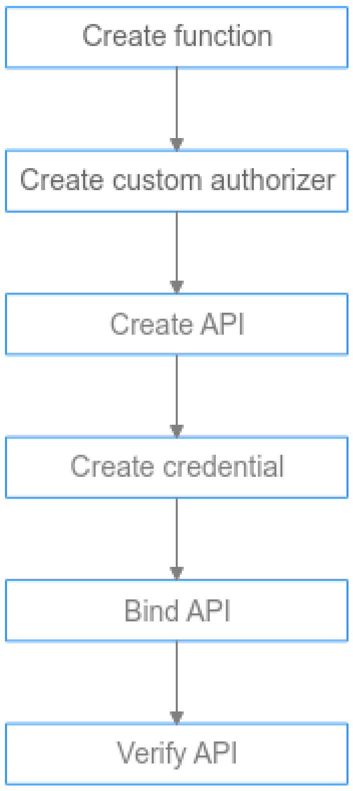
Advantages

In addition to secure app authentication, you can use a custom authorizer to ensure API security.

Restrictions

Custom authentication relies on FunctionGraph.

7.2 General Procedure



1. **Create a function.**

   The function will be used for custom authentication.

2. **Create a custom authorizer.**

   Set the authorizer type to **Frontend**, and select the function created in the previous step.

3. **Create an API.**

   Set authentication mode to **App**, enable **Two-Factor Authentication**, and select the custom authorizer created in the previous step.

4. **Create a credential.**

   APIs that use app authentication require a credential to call. Create a credential to generate an ID and key/secret pair.

5. **Bind the credential to the created API.**

   APIs that use app authentication can be called only with bound credentials.

6. **Verify the API.**

   Call the API to check whether two-factor authentication is configured successfully.

7.3 Implementation Procedure

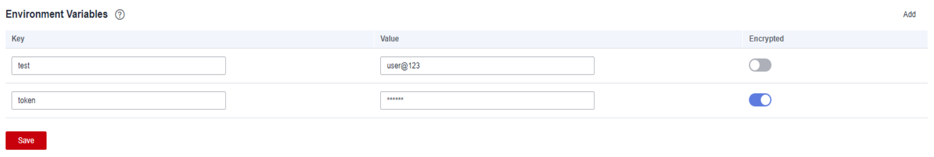
Step 1 Log in to the FunctionGraph console. On the **Dashboard** page, click **Create Function**. For details, see **Developing a Custom Authorizer with FunctionGraph**.

1. Set the parameters according to the following table, and click **Create Function**.

Table 7-1 Function configuration

Parameter	Description
Function Type	Default: **Event Function**
Region	Select the same region as that of APIG.
Function Name	Enter a name that conforms to specific rules to facilitate search.
Agency	An agency that delegates FunctionGraph to access other cloud services. For this example, select **Use no agency**.
Enterprise Project	The default option is **default**.
Runtime	Select **Python 3.9**.

2. On the **Configuration** tab, choose **Environment Variables** in the left pane, and click **Add**. **test** is a header for identity authentication, and **query** is for parameter query. If **token** involves sensitive data, enable the **Encrypted** option.



Environment Variables

Add

Key	Value	Encrypted
test	user@123	Off
token	******	On

Save

3. On the **Code** tab, copy the following code to **index.py**, and click **Deploy**. For details about coding, see section "Creating a Function for Frontend Custom Authentication" in the *API Gateway Developer Guide*.

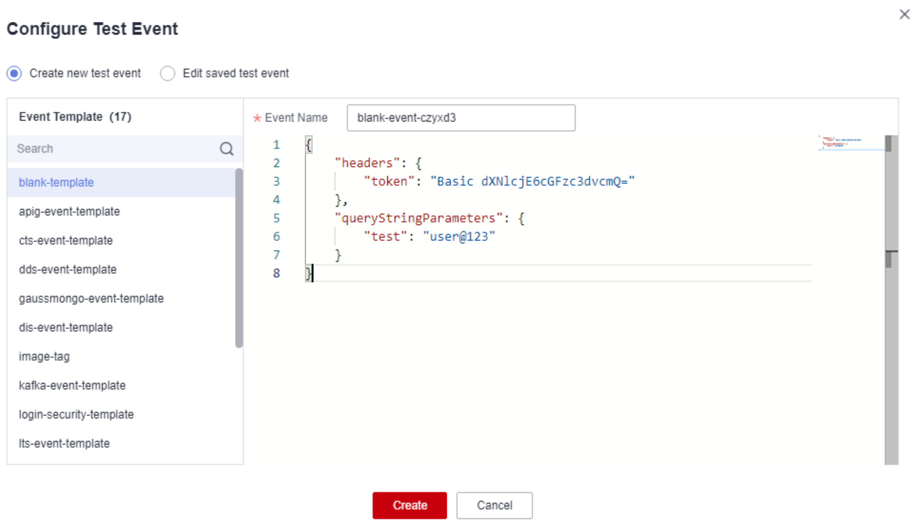
```
# -*- coding:utf-8 -*-
import json
def handler(event, context):
    testParameter = context.getUserData('test');
    userToken = context.getUserData('token');
    if event["headers"].get("token") == userToken and event["queryStringParameters"].get("test") ==
testParameter:
        resp = {
            'statusCode': 200,
            'body': json.dumps({
                "status":"allow",
                "context":{
                    "user":"auth success"
                }
            })
        }
    else:
        resp = {
            'statusCode': 401,
            'body': json.dumps({
                "status":"deny",
            })
        }
    return json.dumps(resp)
```

4. Configure a test event to debug the code.

a. Select **Configure Test Event** from the drop-down list and configure a test event.

NOTE

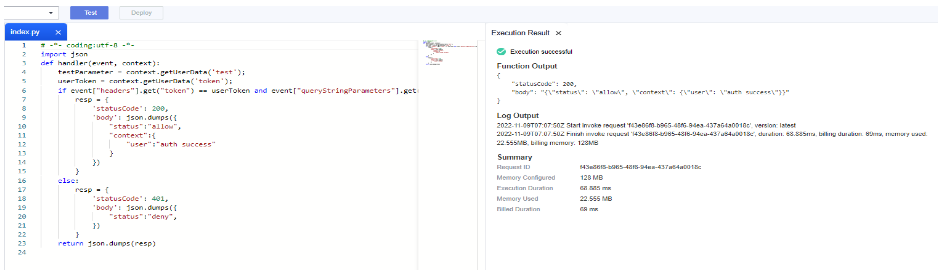
The parameter values in the test event must be the same as those of the environment variables.



Event Name: blank-event-czyxd3

```
{
    "headers": {
        "token": "Basic dXNlcjE6cGFzc3dvcmQ="
    },
    "queryStringParameters": {
        "test": "user@123"
    }
}
```

b. Click **Test**.



```
# -*- coding:utf-8 -*-
import json
def handler(event, context):
    testParameter = context.getUserData('test');
    userToken = context.getUserData('token');
    if event["headers"].get("token") == userToken and event["queryStringParameters"].get
        resp = {
            'statusCode': 200,
            'body': json.dumps({
                "status":"allow",
                "context":{
                    "user":"auth success"
                }
            })
        }
    else:
        resp = {
            'statusCode': 401,
            'body': json.dumps({
                "status":"deny",
            })
        }
    return json.dumps(resp)
```

Execution Result

Execution successful

Function Output

```
{
    "statusCode": 200,
    "body": "{\"status\": \"allow\", \"context\": {\"user\": \"auth success\"}}"
}
```

Log Output

```
2022-11-09T07:07:50Z Start invoke request 'f43e86f8-b965-48f6-94ea-437a64a0018c', version: latest
2022-11-09T07:07:50Z Finish invoke request 'f43e86f8-b965-48f6-94ea-437a64a0018c', duration: 68.885ms, billing duration: 69ms, memory used: 22.555MB, billing memory: 128MB
```

Summary

Request ID	f43e86f8-b965-48f6-94ea-437a64a0018c
Memory Configured	128 MB
Execution Duration	68.885 ms
Memory Used	22.555 MB
Billed Duration	69 ms

c. Click **Deploy**.

Step 2 Log in to the APIG console, and choose **API Management** > **API Policies**.

On the **Custom Authorizers** tab, create a custom authorizer.

Table 7-2 Custom authorizer configuration

Parameter	Description
Name	Enter a name that conforms to specific rules to facilitate search.
Type	Select **Frontend**.
Function URN	Click **Select** and select the **created function**.
Version/Alias	**Version** is selected by default.
Max. Cache Age (s)	30
Identity Sources	Set two identity sources: header **token** and query string **test**.

Step 3 Choose **API Management** > **APIs**, and click **Create API**.

1. Configure the frontend information according to the following table.

Table 7-3 Frontend configuration

Parameter	Description
API Name	Enter a name that conforms to specific rules to facilitate search.
Group	The default option is **DEFAULT**.
URL	**Method**: Request method of the API. Set this parameter to **GET**. **Protocol**: Request protocol of the API. Set this parameter to **HTTPS**. **Subdomain Name**: The system automatically allocates a subdomain name to each API group for internal testing. The subdomain name can be accessed 1000 times a day. **Path**: Path for requesting the API. Enter **/api/two_factor_authorization**.
Gateway Response	Select a response to be displayed if the gateway fails to process an API request. The default gateway response is **default**.
Authentication Mode	API authentication mode. Set this parameter to **App**.
Two-Factor Authentication	Enable this option and select a custom authorizer.

2. Click **Next** and set the backend type to **Mock**.

   Select a status code, set the response, and click **Finish**.
3. Publish the API.

Step 4 In the navigation pane, choose **API Management** > **Credentials**.

Click **Create Credential**, enter a credential name, and click **OK**.

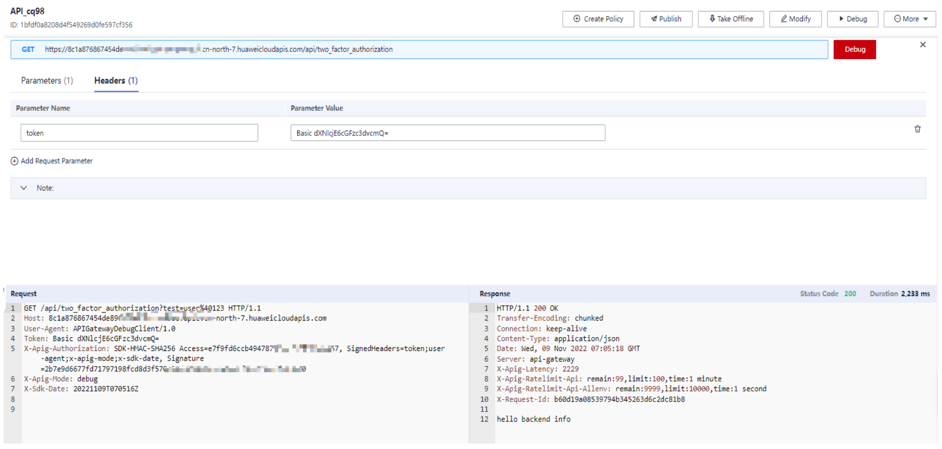
Step 5 Bind this credential to the API.

Click the credential name to go to the details page. In the **APIs** area, click **Bind to APIs**, select an API, and click **OK**.

Step 6 Verify the API.

- Call the API on the debugging page of APIG to verify if two-factor authentication is configured successfully.

  Add **test** on the **Parameters** tab and add **token** on the **Headers** tab. Use the same parameter values set for the custom authentication function. If the parameter values are different, the server will return a 401 message indicating that the authentication fails.



GET https://8c1a876867454de[redacted].cn-north-7.huaweicloudapis.com/api/two_factor_authorization

Parameter Name	Parameter Value
token	Basic dXNlcjE6cGFzc3dvcmQ=

Request

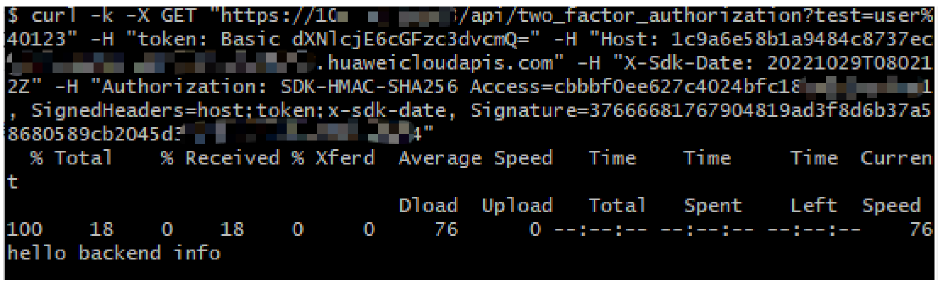
```
GET /api/two_factor_authorization?test=user%40123 HTTP/1.1
Host: 8c1a876867454de89[redacted]north-7.huaweicloudapis.com
User-Agent: APIGatewayDebugClient/1.0
Token: Basic dXNlcjE6cGFzc3dvcmQ=
X-Apig-Authorization: SDK-HMAC-SHA256 Access=e7f9fd6ccb494787[redacted]67, SignedHeaders=token;user
    -agent;x-apig-mode;x-sdk-date, Signature
    =2b7e9d6677fd71797198fcd8d3f57[redacted]0
X-Apig-Mode: debug
X-Sdk-Date: 20221109T070516Z
```

Response Status Code 200 Duration 2,233 ms

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
Connection: keep-alive
Content-Type: application/json
Date: Wed, 09 Nov 2022 07:05:18 GMT
Server: api-gateway
X-Apig-Latency: 2229
X-Apig-Ratelimit-Api: remain:99,limit:100,time:1 minute
X-Apig-Ratelimit-Api-Allenv: remain:9999,limit:10000,time:1 second
X-Request-Id: b60d19a08539794b345263d6c2dc81b8

hello backend info
```

- Alternatively, call the API with a **curl** command. Download the JavaScript SDK first. To call the API, input a key and secret as well as the header and query string to generate a **curl** command, and then copy this command to your CLI for execution. For details, see section "curl" in the *API Gateway Developer Guide*.



```
$ curl -k -X GET "https://1C[redacted]/api/two_factor_authorization?test=user%
40123" -H "token: Basic dXNlcjE6cGFzc3dvcmQ=" -H "Host: 1c9a6e58b1a9484c8737ec
[redacted].huaweicloudapis.com" -H "X-Sdk-Date: 20221029T08021
2Z" -H "Authorization: SDK-HMAC-SHA256 Access=cbbbf0ee627c4024bfc18[redacted]1
, SignedHeaders=host;token;x-sdk-date, Signature=37666681767904819ad3f8d6b37a5
8680589cb2045d3[redacted]4"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Curren
t
                                 Dload  Upload   Total   Spent    Left  Speed
100    18    0    18    0     0     76      0 --:--:-- --:--:-- --:--:--    76
hello backend info
```

----End

8 Change History

Table 8-1 Change history

Date	Description
2023-04-30	This issue incorporates the following changes: ● Updated this document for the new console. ● Added **Request Throttling 2.0** and **Two-Factor Authentication**.
2023-04-12	This issue incorporates the following change: Added **Exposing Backend Services Across VPCs** and **Interconnecting with WAF**.
2021-09-30	This issue is the first official release.