Anti-DDoS

API Reference

Issue 02

Date 2025-10-09

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are the property 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 Cloud 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.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: https://www.huaweicloud.com/intl/en-us/

Contents

1 Before You Start

Overview

The Anti-DDoS service protects public IP addresses against Layer 4 to Layer 7 distributed denial of service (DDoS) attacks and sends alarms immediately when detecting an attack. Anti-DDoS improves the bandwidth utilization and ensures the stable running of user services.

This document describes how to use application programming interfaces (APIs) to perform operations to Anti-DDoS, such as querying or updating Anti-DDoS protection policy. For details about all supported operations, see **API Overview**.

Before calling Anti-DDoS APIs, ensure that you are familiar with Anti-DDoS concepts. For details, see section "Service Overview" of the *Anti-DDoS User Guide*.

API Calling

Anti-DDoS provides Representational State Transfer (REST) APIs, allowing you to use HTTPS requests to call them. For details, see **API Calling**.

Notes and Constraints

For details about the constraints, see the API description.

2 API Overview

You can use all functions of Anti-DDoS through its APIs.

Type	Description
Protection Management	These APIs are used to enable and update the Anti-DDoS service, and query Anti-DDoS running information, including EIP defense status, defense statistics, defense traffic, and abnormal events.
Task Management	These APIs are used to query Anti-DDoS tasks.
Alarm Configuration Management	These APIs are used to query and update Anti-DDoS alarm notification configurations.

3 API Calling

3.1 Making an API Request

This section describes the structure of a REST API request, and uses the IAM API for obtaining a user token as an example to demonstrate how to call an API. The obtained token can then be used to authenticate the calling of other APIs.

Request URI

A request URI is in the following format:

{URI-scheme} :// {Endpoint} / {resource-path} ? {query-string}

Although a request URI is included in the request header, most programming languages or frameworks require the request URI to be transmitted separately.

- **URI-scheme**:

  Protocol used to transmit requests. All APIs use HTTPS.
- **Endpoint**:

  Domain name or IP address of the server bearing the REST service. The endpoint varies between services in different regions. It can be obtained from the administrator.
- **resource-path**:

  Access path of an API for performing a specified operation. Obtain the path from the URI of an API. For example, the **resource-path** of the API used to obtain a user token is **/v3/auth/tokens**.
- **query-string**:

  Query parameter, which is optional. Ensure that a question mark (?) is included before each query parameter that is in the format of "Parameter name=Parameter value". For example, **?limit=10** indicates that a maximum of 10 data records will be displayed.

NOTE

To simplify the URI display in this document, each API is provided only with a **resource-path** and a request method. The **URI-scheme** of all APIs is **HTTPS**, and the endpoints of all APIs in the same region are identical.

Request Methods

The HTTP protocol defines the following request methods that can be used to send a request to the server:

- **GET**: requests the server to return specified resources.
- **PUT**: requests the server to update specified resources.
- **POST**: requests the server to add resources or perform special operations.
- **DELETE**: requests the server to delete specified resources, for example, an object.
- **HEAD**: same as GET except that the server must return only the response header.
- **PATCH**: requests the server to update partial content of a specified resource. If the resource does not exist, a new resource will be created.

For example, in the case of the API used to obtain a user token, the request method is POST. The request is as follows:

```
POST https://iam.ap-southeast-1.myhuaweicloud.com/v3/auth/tokens
```

Request Header

You can also add additional header fields to a request, such as the fields required by a specified URI or HTTP method. For example, to request for the authentication information, add **Content-Type**, which specifies the request body type.

Common request header fields are as follows:

- **Content-Type**: specifies the request body type or format. This field is mandatory and its default value is **application/json**. Other values of this field will be provided for specific APIs if any.
- **X-Auth-Token**: specifies a user token only for token-based API authentication. The user token is a response to the API used to obtain a user token. This API is the only one that does not require authentication.

  NOTE

  In addition to supporting token-based authentication, APIs also support authentication using access key ID/secret access key (AK/SK). During AK/SK-based authentication, an SDK is used to sign the request, and the **Authorization** (signature information) and **X-Sdk-Date** (time when the request is sent) header fields are automatically added to the request.

  For more information, see **AK/SK-based Authentication**.

The API used to obtain a user token does not require authentication. Therefore, only the **Content-Type** field needs to be added to requests for calling the API. An example of such requests is as follows:

```
POST https://iam.ap-southeast-1.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
```

Request Body

The body of a request is often sent in a structured format as specified in the **Content-Type** header field. The request body transfers content except the request header.

The request body varies between APIs. Some APIs do not require the request body, such as the APIs requested using the GET and DELETE methods.

In the case of the API used to obtain a user token, the request parameters and parameter description can be obtained from the API request. The following provides an example request with a body included. Set ***username*** to the name of a user, ***domainname*** to the name of the account that the user belongs to, ******** to the user's login password, and ***xxxxxxxxxxxxxxxxxx*** to the project name. You can learn more information about projects from the administrator.

NOTE

The **scope** parameter specifies where a token takes effect. You can set **scope** to an account or a project under an account. In the following example, the token takes effect only for the resources in a specified project. For more information about this API, see "Obtaining a User Token".

```
POST https://iam.ap-southeast-1.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "username",
                    "password": "********",
                    "domain": {
                        "name": "domainname"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "name": "xxxxxxxxxxxxxxxxxx"
            }
        }
    }
}
```

If all data required for the API request is available, you can send the request to call the API through **curl**, **Postman**, or coding. In the response to the API used to obtain a user token, **x-subject-token** is the desired user token. This token can then be used to authenticate the calling of other APIs.

3.2 Authentication

Requests for calling an API can be authenticated using either of the following methods:

- Token-based authentication: Requests are authenticated using a token.
- AK/SK-based authentication: Requests are authenticated by encrypting the request body using an AK/SK pair. This method is recommended because it provides higher security than token-based authentication.

Token-based Authentication

NOTE

The validity period of a token is 24 hours. When using a token for authentication, cache it to prevent frequently calling the IAM API used to obtain a user token.

A token specifies temporary permissions in a computer system. During API authentication using a token, the token is added to requests to get permissions for calling the API.

The token can be obtained by calling the required API. For more information, see Obtaining a User Token. A project-level token is required for calling this API, that is, **auth.scope** must be set to **project** in the request body. Example:

```
{
   "auth": {
      "identity": {
         "methods": [
            "password"
         ],
         "password": {
            "user": {
               "name": "username",
               "password": "********",
               "domain": {
                  "name": "domainname"
               }
            }
         }
      },
      "scope": {
         "project": {
            "name": "xxxxxxxx"
         }
      }
   }
}
```

After a token is obtained, the **X-Auth-Token** header field must be added to requests to specify the token when calling other APIs. For example, if the token is **ABCDEFJ....**, **X-Auth-Token: ABCDEFJ....** can be added to a request as follows:

```
POST https://iam.ap-southeast-1.myhuaweicloud.com/v3/auth/projects
Content-Type: application/json
X-Auth-Token: ABCDEFJ....
```

AK/SK-based Authentication

NOTE

AK/SK-based authentication supports API requests with a body not larger than 12 MB. For API requests with a larger body, token-based authentication is recommended.

In AK/SK-based authentication, AK/SK is used to sign requests and the signature is then added to the requests for authentication.

- AK: access key ID, which is a unique identifier used in conjunction with a secret access key to sign requests cryptographically.
- SK: secret access key used in conjunction with an AK to sign requests cryptographically. It identifies a request sender and prevents the request from being modified.

In AK/SK-based authentication, you can use an AK/SK to sign requests based on the signature algorithm or use the signing SDK to sign requests.

NOTICE

The signing SDK is only used for signing requests and is different from the SDKs provided by services.

3.3 Response

Status Code

After sending a request, you will receive a response, including a status code, response header, and response body.

A status code is a group of digits, ranging from 1xx to 5xx. It indicates the status of a request. For more information, see **Status Code**.

For example, if status code **201** is returned for calling the API used to obtain a user token, the request is successful.

Response Header

Similar to a request, a response also has a header, for example, **Content-Type**.

The following shows the response header for the API to obtain a user token, in which **x-subject-token** is the desired user token. This token can then be used to authenticate the calling of other APIs.

Figure 3-1 Header fields of the response to the request for obtaining a user token

(Optional) Response Body

The body of a response is often returned in structured format as specified in the **Content-Type** header field. The response body transfers content except the response header.

The following shows part of the response body for the API to obtain a user token. For the sake of space, only part of the content is displayed here.

```
{
    "token": {
        "expires_at": "2019-02-13T06:52:13.855000Z",
        "methods": [
            "password"
        ],
        "catalog": [
            {
                "endpoints": [
                    {
                        "region_id": "xxxxxxxx",
.....
```

If an error occurs during API calling, an error code and a message will be displayed. The following shows an error response body.

```
{
    "error_msg": "The format of message is error",
    "error_code": "AS.0001"
}
```

In the response body, **error_code** is an error code, and **error_msg** provides information about the error.

4 API

4.1 Anti-DDoS Task Management

4.1.1 Querying Anti-DDoS Tasks

Function

This API enables you to query the execution status of a specified Anti-DDoS configuration task.

Calling Method

For details, see **Calling APIs**.

URI

GET /v2/{project_id}/query-task-status

Table 4-1 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.

Table 4-2 Query Parameters

Parameter	Mandatory	Type	Description
task_id	Yes	String	Task ID (non-negative integer) string.

Request Parameters

Table 4-3 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-4 Response body parameters

Parameter	Type	Description
task_status	String	Task status. The options are as follows: • **success**: successful • **failed**: failed • **waiting**: waiting • **running**: running • **preprocess**: preprocessing • **ready**: ready
task_msg	String	Additional information about a task.

Example Requests

None

Example Responses

Status code: 200

Request succeeded.

```
{
  "task_status" : "success",
  "task_msg" : ""
}
```


Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.2 DDoS Protection Management

4.2.1 Querying the List of EIP Defense Statuses

Function

Query the defense statuses of all EIPs, regardless of whether an EIP has been bound to an ECS or not.

Calling Method

For details, see **Calling APIs**.

URI

GET /v1/{project_id}/antiddos

Table 4-5 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.

Table 4-6 Query Parameters

Parameter	Mandatory	Type	Description
status	No	String	Values: • **normal**: normal • **configging**: configuration in progress • **notConfig**: not configured • **packetcleaning**: cleaning • **packetdropping**: blackhole If this parameter is not specified, the defense statuses of all ECSs are displayed in the Neutron-queried order by default.
limit	No	String	Maximum number of returned results. The value ranges from 1 to 100.
offset	No	String	Offset. The value ranges from 0 to 2147483647.
ip	No	String	IP address, which can be in IPv4 or IPv6 format. Partial query is supported. For example, if you enter **?ip=192.168**, the EIP protection status of 192.168.111.1 and 10.192.168.8 will be returned.

Request Parameters

Table 4-7 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-8 Response body parameters

Parameter	Type	Description
total	Integer	Total number of EIPs.
ddosStatus	Array of DDosStatus objects	Protection status list.

Table 4-9 DDosStatus

Parameter	Type	Description
floating_ip_id	String	EIP ID.
floating_ip_address	String	Floating IP address.
network_type	String	EIP type. The value can be: • **EIP**: EIP that is bound or not bound with ECS. • **ELB**: EIP that is bound with ELB.
status	String	Protection status. The options are as follows: • **normal**: normal • **configging**: configuration in progress • **notConfig**: not configured • **packetcleaning**: cleaning • **packetdropping**: blackhole
blackhole_endtime	Long	End time of a black hole.
protect_type	String	Protection type.
traffic_threshold	Long	Traffic threshold.
http_threshold	Long	HTTP traffic threshold.

Example Requests

None

Example Responses

Status code: 200

Request succeeded.


```
{
  "total" : 1,
  "ddosStatus" : [ {
    "floating_ip_id" : "18e6ace5-eb36-4196-a15e-1e000c24e026",
    "floating_ip_address" : "139.9.116.167",
    "network_type" : "EIP",
    "status" : "normal",
    "blackhole_endtime" : 0,
    "protect_type" : "default",
    "traffic_threshold" : 99,
    "http_threshold" : 0
  } ]
}
```


Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.2.2 Querying Anti-DDoS Specifications

Function

This API allows you to query available Anti-DDoS defense policies. You can select a policy for Anti-DDoS traffic cleaning.

Calling Method

For details, see **Calling APIs**.

URI

GET /v2/{project_id}/antiddos/query-config-list

Table 4-10 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.

Request Parameters

Table 4-11 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-12 Response body parameters

Parameter	Type	Description
traffic_limited_list	Array of **TriggerBpsDict** objects	List of traffic limits.
http_limited_list	Array of **TriggerQpsDict** objects	List of HTTP limits.
connection_limited_list	Array of **CleanLimitDict** objects	List of connection limits.
extend_ddos_config	Array of **ExtendDDoSSet** objects	Extended configuration list.

Table 4-13 TriggerBpsDict

Parameter	Type	Description
traffic_pos_id	Long	Position ID of traffic. The options are as follows: **1**: 10 Mbit/s; **2**: 30 Mbit/s; **3**: 50 Mbit/s; **4**: 70 Mbit/s; **5**: 100 Mbit/s; **6**: 150 Mbit/s; **7**: 200 Mbit/s; **8**: 250 Mbit/s; **9**: 300 Mbit/s; **10**: 500 Mbit/s; **11**: 800 Mbit/s;**88**: 1,000 Mbit/s; **99**: default protection

Parameter	Type	Description
traffic_per_second	Long	Threshold of traffic per second (Mbit/s).
packet_per_second	Long	Threshold of number of packets per second.

Table 4-14 TriggerQpsDict

Parameter	Type	Description
http_request_pos_id	Long	Segment ID of the HTTP request counts. The value is fixed to **1**.
http_packet_per_second	Long	Threshold of number of HTTP requests per second.

Table 4-15 CleanLimitDict

Parameter	Type	Description
cleaning_access_pos_id	Long	Position ID of access limit during cleaning. The options are as follows: **1**: 10 Mbit/s; **2**: 30 Mbit/s; **3**: 50 Mbit/s; **4**: 70 Mbit/s; **5**: 100 Mbit/s; **6**: 150 Mbit/s; **7**: 200 Mbit/s; **8**: 250 Mbit/s; **9**: 300 Mbit/s; **10**: 500 Mbit/s; **11**: 800 Mbit/s;**88**: 1,000 Mbit/s; **99**: default protection
new_connection_limited	Long	Number of new connections of a source IP address.
total_connection_limited	Long	Total number of connections of a source IP address.

Table 4-16 ExtendDDoSSet

Parameter	Type	Description
SetID	Long	Configuration segment ID.
new_connection_limited	Long	Number of new connections of a source IP address.
total_connection_limited	Long	Total number of connections of a source IP address.

Parameter	Type	Description
http_packet_per_second	Long	Threshold of number of HTTP requests per second.
traffic_per_second	Long	Threshold of traffic per second (Mbit/s).
packet_per_second	Long	Threshold of number of packets per second.

Example Requests

None

Example Responses

Status code: 200

Request succeeded.


```
{
  "traffic_limited_list" : [ {
    "traffic_pos_id" : 1,
    "traffic_per_second" : 10,
    "packet_per_second" : 2000
  }, {
    "traffic_pos_id" : 2,
    "traffic_per_second" : 30,
    "packet_per_second" : 6000
  }, {
    "traffic_pos_id" : 3,
    "traffic_per_second" : 50,
    "packet_per_second" : 10000
  }, {
    "traffic_pos_id" : 4,
    "traffic_per_second" : 70,
    "packet_per_second" : 15000
  }, {
    "traffic_pos_id" : 5,
    "traffic_per_second" : 100,
    "packet_per_second" : 20000
  }, {
    "traffic_pos_id" : 6,
    "traffic_per_second" : 150,
    "packet_per_second" : 25000
  }, {
    "traffic_pos_id" : 7,
    "traffic_per_second" : 200,
    "packet_per_second" : 35000
  }, {
    "traffic_pos_id" : 8,
    "traffic_per_second" : 250,
    "packet_per_second" : 50000
  }, {
    "traffic_pos_id" : 9,
    "traffic_per_second" : 300,
    "packet_per_second" : 70000
  }, {
    "traffic_pos_id" : 88,
    "traffic_per_second" : 1000,
    "packet_per_second" : 300000
```



```
} ],
"http_limited_list" : [ {
  "http_request_pos_id" : 1,
  "http_packet_per_second" : 100
}, {
  "http_request_pos_id" : 2,
  "http_packet_per_second" : 150
}, {
  "http_request_pos_id" : 3,
  "http_packet_per_second" : 240
}, {
  "http_request_pos_id" : 4,
  "http_packet_per_second" : 350
}, {
  "http_request_pos_id" : 5,
  "http_packet_per_second" : 480
}, {
  "http_request_pos_id" : 6,
  "http_packet_per_second" : 550
}, {
  "http_request_pos_id" : 7,
  "http_packet_per_second" : 700
}, {
  "http_request_pos_id" : 8,
  "http_packet_per_second" : 850
}, {
  "http_request_pos_id" : 9,
  "http_packet_per_second" : 1000
}, {
  "http_request_pos_id" : 10,
  "http_packet_per_second" : 1500
}, {
  "http_request_pos_id" : 11,
  "http_packet_per_second" : 2000
}, {
  "http_request_pos_id" : 12,
  "http_packet_per_second" : 3000
}, {
  "http_request_pos_id" : 13,
  "http_packet_per_second" : 5000
}, {
  "http_request_pos_id" : 14,
  "http_packet_per_second" : 10000
}, {
  "http_request_pos_id" : 15,
  "http_packet_per_second" : 20000
} ],
"connection_limited_list" : [ {
  "cleaning_access_pos_id" : 1,
  "new_connection_limited" : 10,
  "total_connection_limited" : 30
}, {
  "cleaning_access_pos_id" : 2,
  "new_connection_limited" : 20,
  "total_connection_limited" : 100
}, {
  "cleaning_access_pos_id" : 3,
  "new_connection_limited" : 30,
  "total_connection_limited" : 200
}, {
  "cleaning_access_pos_id" : 4,
  "new_connection_limited" : 40,
  "total_connection_limited" : 250
}, {
  "cleaning_access_pos_id" : 5,
  "new_connection_limited" : 50,
  "total_connection_limited" : 300
}, {
  "cleaning_access_pos_id" : 6,
```



```
    "new_connection_limited" : 60,
    "total_connection_limited" : 500
  }, {
    "cleaning_access_pos_id" : 7,
    "new_connection_limited" : 70,
    "total_connection_limited" : 600
  }, {
    "cleaning_access_pos_id" : 8,
    "new_connection_limited" : 80,
    "total_connection_limited" : 700
  } ],
  "extend_ddos_config" : [ ]
}
```


Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.2.3 Querying Weekly Defense Statistics

Function

This API allows you to query weekly defense statistics about all your EIPs, including the number of blocked DDoS attacks, number of attacks, and ranking by the number of attacks. Currently, you can query weekly statistics up to four weeks before the current time. Data older than four weeks cannot be queried.

Calling Method

For details, see **Calling APIs**.

URI

GET /v1/{project_id}/antiddos/weekly

Table 4-17 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.

Table 4-18 Query Parameters

Parameter	Mandatory	Type	Description
period_start_date	No	String	Start date of a week.

Request Parameters

Table 4-19 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-20 Response body parameters

Parameter	Type	Description
ddos_intercept_times	Integer	Number of DDoS attacks blocked in a week.
weekdata	Array of **WeeklyCount** objects	Number of attacks in a week.
top10	Array of **WeeklyTop10** objects	Top 10 attacked IP addresses.

Table 4-21 WeeklyCount

Parameter	Type	Description
ddos_intercept_times	Integer	Number of DDoS attacks blocked.

Parameter	Type	Description
ddos_blackhole_times	Integer	Number of DDoS black holes.
max_attack_bps	Integer	Maximum attack traffic.
max_attack_conns	Integer	Maximum number of attack connections.
period_start_date	Long	Start time.

Table 4-22 WeeklyTop10

Parameter	Type	Description
floating_ip_address	String	EIP.
times	Integer	Number of DDoS attacks blocked by cleaning operations and black holes.

Example Requests

None

Example Responses

Status code: 200

Request succeeded.


```
{
  "ddos_intercept_times" : 0,
  "weekdata" : [ {
    "ddos_intercept_times" : 0,
    "ddos_blackhole_times" : 0,
    "max_attack_bps" : 0,
    "max_attack_conns" : 0,
    "period_start_date" : 1605496722606
  }, {
    "ddos_intercept_times" : 0,
    "ddos_blackhole_times" : 0,
    "max_attack_bps" : 0,
    "max_attack_conns" : 0,
    "period_start_date" : 1605583122606
  }, {
    "ddos_intercept_times" : 0,
    "ddos_blackhole_times" : 0,
    "max_attack_bps" : 0,
    "max_attack_conns" : 0,
    "period_start_date" : 1605669522606
  }, {
    "ddos_intercept_times" : 0,
    "ddos_blackhole_times" : 0,
    "max_attack_bps" : 0,
    "max_attack_conns" : 0,
    "period_start_date" : 1605755922606
```



```
 }, {
   "ddos_intercept_times" : 0,
   "ddos_blackhole_times" : 0,
   "max_attack_bps" : 0,
   "max_attack_conns" : 0,
   "period_start_date" : 1605842322606
 }, {
   "ddos_intercept_times" : 0,
   "ddos_blackhole_times" : 0,
   "max_attack_bps" : 0,
   "max_attack_conns" : 0,
   "period_start_date" : 1605928722606
 }, {
   "ddos_intercept_times" : 0,
   "ddos_blackhole_times" : 0,
   "max_attack_bps" : 0,
   "max_attack_conns" : 0,
   "period_start_date" : 1606015122606
 } ],
 "top10" : [ ]
}
```


Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.2.4 Querying Anti-DDoS Policies

Function

This API enables you to query configured Anti-DDoS defense policies. You can query the policy of a specified EIP.

Calling Method

For details, see **Calling APIs**.

URI

GET /v1/{project_id}/antiddos/{floating_ip_id}

Table 4-23 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.
floating_ip_id	Yes	String	ID corresponding to the EIP of a user.

Table 4-24 Query Parameters

Parameter	Mandatory	Type	Description
ip	No	String	EIP of a user.

Request Parameters

Table 4-25 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-26 Response body parameters

Parameter	Type	Description
enable_L7	Boolean	Whether to enable layer-7 protection. The value is fixed to **false**.
traffic_pos_id	Long	Position ID of traffic. The options are as follows: **1**: 10 Mbit/s; **2**: 30 Mbit/s; **3**: 50 Mbit/s; **4**: 70 Mbit/s; **5**: 100 Mbit/s; **6**: 150 Mbit/s; **7**: 200 Mbit/s; **8**: 250 Mbit/s; **9**: 300 Mbit/s; **10**: 500 Mbit/s; **11**: 800 Mbit/s;**88**: 1,000 Mbit/s; **99**: default protection
http_request_pos_id	Long	Segment ID of the HTTP request counts. The value is fixed to **1**.

Parameter	Type	Description
cleaning_access_pos_id	Long	Position ID of access limit during cleaning. The options are as follows: **1**: 10 Mbit/s; **2**: 30 Mbit/s; **3**: 50 Mbit/s; **4**: 70 Mbit/s; **5**: 100 Mbit/s; **6**: 150 Mbit/s; **7**: 200 Mbit/s; **8**: 250 Mbit/s; **9**: 300 Mbit/s; **10**: 500 Mbit/s; **11**: 800 Mbit/s;**88**: 1,000 Mbit/s; **99**: default protection
app_type_id	Long	Application type ID. The value is fixed to **0**.
antiddos_config_name	String	Protection level name.
antiddos_config_id	String	Protection level ID.

Example Requests

None

Example Responses

Status code: 200

Request succeeded.

```
{
  "enable_L7" : false,
  "traffic_pos_id" : 8,
  "http_request_pos_id" : 8,
  "cleaning_access_pos_id" : 8,
  "app_type_id" : 1,
  "antiddos_config_name" : "test",
  "antiddos_config_id" : "1234567890"
}
```

Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.2.5 Updating the Anti-DDoS Protection

Function

This API enables you to update the Anti-DDoS defense policy of a specified EIP. If this API is successfully called, it indicates that the service node receives the request for updating the Anti-DDoS policy of the EIP. To check whether the operation is successful, call the task query API to query the task execution status. For details, see "Querying Anti-DDoS Tasks".

Calling Method

For details, see **Calling APIs**.

URI

PUT /v1/{project_id}/antiddos/{floating_ip_id}

Table 4-27 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.
floating_ip_id	Yes	String	ID corresponding to the EIP of a user.

Table 4-28 Query Parameters

Parameter	Mandatory	Type	Description
ip	No	String	ip

Request Parameters

Table 4-29 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Table 4-30 Request body parameters

Parameter	Mandatory	Type	Description
app_type_id	Yes	Integer	Application type ID. The value is fixed to **0**.
cleaning_access_pos_id	Yes	Integer	Position ID of access limit during cleaning. The options are as follows: **1**: 10 Mbit/s; **2**: 30 Mbit/s; **3**: 50 Mbit/s; **4**: 70 Mbit/s; **5**: 100 Mbit/s; **6**: 150 Mbit/s; **7**: 200 Mbit/s; **8**: 250 Mbit/s; **9**: 300 Mbit/s; **10**: 500 Mbit/s; **11**: 800 Mbit/s;**88**: 1,000 Mbit/s; **99**: default protection
enable_L7	Yes	Boolean	Whether to enable layer-7 protection. The value is fixed to **false**.
http_request_pos_id	Yes	Integer	Segment ID of the HTTP request counts. The value is fixed to **1**.
traffic_pos_id	Yes	Integer	Position ID of traffic. The options are as follows: **1**: 10 Mbit/s; **2**: 30 Mbit/s; **3**: 50 Mbit/s; **4**: 70 Mbit/s; **5**: 100 Mbit/s; **6**: 150 Mbit/s; **7**: 200 Mbit/s; **8**: 250 Mbit/s; **9**: 300 Mbit/s; **10**: 500 Mbit/s; **11**: 800 Mbit/s;**88**: 1,000 Mbit/s; **99**: default protection
antiddos_config_id	No	String	Protection level ID.

Response Parameters

Status code: 200

Table 4-31 Response body parameters

Parameter	Type	Description
error_code	String	Internal error code.
error_msg	String	Internal error description.

Parameter	Type	Description
task_id	String	ID of a task. This ID can be used to query the status of the task. This field is reserved for future task audit extension and is not required currently.

Example Requests

Update the Anti-DDoS policy for a specified EIP. Set the position ID of access limit during cleaning to 8, and set the position ID of traffic to 1.

```
PUT https://{endpoint}/v1/{project_id}/antiddos/{floating_ip_id}

{
  "app_type_id" : 0,
  "cleaning_access_pos_id" : 8,
  "enable_L7" : false,
  "http_request_pos_id" : 1,
  "traffic_pos_id" : 1,
  "antiddos_config_name" : "test",
  "antiddos_config_id" : "1234567890"
}
```

Example Responses

Status code: 200

Request succeeded.

```
{
  "error_code" : "10000000",
  "error_msg" : "The task has been received and is being handled",
  "task_id" : "59385d2a-6266-4d3a-9122-a228c530f557"
}
```

Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.2.6 Querying the Traffic of a Specified EIP

Function

You can query the traffic of a specified EIP in the last 24 hours. Traffic is detected at five-minute intervals.

Calling Method

For details, see **Calling APIs**.

URI

GET /v1/{project_id}/antiddos/{floating_ip_id}/daily

Table 4-32 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.
floating_ip_id	Yes	String	ID corresponding to the EIP of a user.

Table 4-33 Query Parameters

Parameter	Mandatory	Type	Description
ip	No	String	EIP of a user.

Request Parameters

Table 4-34 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-35 Response body parameters

Parameter	Type	Description
data	Array of **DailyData** objects	Traffic in the last 24 hours.

Table 4-36 DailyData

Parameter	Type	Description
period_start	Long	Start time.
bps_in	Integer	Ingress traffic (bit/s).
bps_attack	Long	Attack traffic (bit/s).
total_bps	Long	Total traffic.
pps_in	Long	Number of inbound packets per second.
pps_attack	Long	Number of attack packets per second.
total_pps	Long	Total inbound packet rate.

Example Requests

None

Example Responses

Status code: 200

Request succeeded.

```
{
  "data" : [ {
    "period_start" : 1606188642720,
    "bps_in" : 0,
    "bps_attack" : 0,
    "total_bps" : 0,
    "pps_in" : 0,
    "pps_attack" : 0,
    "total_pps" : 0
  } ]
}
```

Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.2.7 Querying Events of a Specified EIP

Function

This API allows you to query events of a specified EIP in the last 24 hours. Events include cleaning and black hole events. The query delay is no more than 5 minutes.

Calling Method

For details, see **Calling APIs**.

URI

GET /v1/{project_id}/antiddos/{floating_ip_id}/logs

Table 4-37 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.
floating_ip_id	Yes	String	ID corresponding to the EIP of a user.

Table 4-38 Query Parameters

Parameter	Mandatory	Type	Description
sort_dir	No	String	Values: • **desc**: descending order of time • **asc**: ascending order of time The default value is **desc**.
limit	No	String	Maximum number of returned results. The value ranges from 1 to 100. This parameter is used together with **offset**. If neither **limit** nor **offset** is specified, all servers are returned.
offset	No	String	Offset. This field is valid only if **limit** is specified.
ip	No	String	EIP of a user.

Request Parameters

Table 4-39 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-40 Response body parameters

Parameter	Type	Description
total	Long	Total number of EIPs.
logs	Array of **DailyLog** objects	Abnormal event list.

Table 4-41 DailyLog

Parameter	Type	Description
start_time	Long	Start time.
end_time	Long	End time.
status	Integer	Protection status. The options are as follows: • **1**: cleaning • **2**: blackhole
trigger_bps	Integer	Traffic at the triggering point.
trigger_pps	Integer	Packet rate at the triggering point.
trigger_http_pps	Integer	HTTP request rate at the triggering point.

Example Requests

None

Example Responses

Status code: 200

Request succeeded.

```
{
  "total" : 0,
  "logs" : [ ]
}
```

Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.2.8 Querying the Defense Status of a Specified EIP

Function

This API is used to query the defense status of a specified EIP.

Calling Method

For details, see **Calling APIs**.

URI

GET /v1/{project_id}/antiddos/{floating_ip_id}/status

Table 4-42 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.
floating_ip_id	Yes	String	ID corresponding to the EIP of a user.

Table 4-43 Query Parameters

Parameter	Mandatory	Type	Description
ip	No	String	EIP of a user.

Request Parameters

Table 4-44 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-45 Response body parameters

Parameter	Type	Description
status	String	Protection status. The options are as follows: • **normal**: normal • **configging**: configuration in progress • **notConfig**: not configured • **packetcleaning**: cleaning • **packetdropping**: blackhole

Example Requests

None

Example Responses

Status code: 200

Request succeeded.


```
{
  "status" : "normal"
}
```

Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.2.9 Querying the List of a User's EIP Defense Statuses

Function

This API enables you to query the defense statuses of all EIPs, regardless whether an EIP has been bound to an Elastic Cloud Server (ECS) or not.

Calling Method

For details, see **Calling APIs**.

URI

GET /v2/{project_id}/antiddos

Table 4-46 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.

Table 4-47 Query Parameters

Parameter	Mandatory	Type	Description
status	No	String	Values: • **normal**: normal • **configging**: configuration in progress • **notConfig**: not configured • **packetcleaning**: cleaning • **packetdropping**: blackhole If this parameter is not specified, the defense statuses of all ECSs are displayed in the Neutron-queried order by default.
limit	No	String	Maximum number of returned results. The value ranges from 1 to 100.
offset	No	String	Offset. The value ranges from 0 to 2147483647.
ips	No	String	IP address, which can be in IPv4 or IPv6 format. Partial query is supported. For example, if you enter **?ip=192.168**, the EIP protection status of 192.168.111.1 and 10.192.168.8 will be returned.

Request Parameters

Table 4-48 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-49 Response body parameters

Parameter	Type	Description
total	Integer	Total number of EIPs.
ddosStatus	Array of **EipDDosStatus** objects	Protection status list.

Table 4-50 EipDDosStatus

Parameter	Type	Description
floating_ip_id	String	EIP ID.
floating_ip_address	String	Floating IP address.
product_type	String	EIP type. The value can be: • **Anti-DDoS**: The EIP is protected by Anti-DDoS. • **CNAD**: The EIP is protected by Cloud Native Anti-DDoS (CNAD).
status	String	Protection status. The options are as follows: • **normal**: normal • **configging**: configuration in progress • **notConfig**: not configured • **packetcleaning**: cleaning • **packetdropping**: blackhole
clean_threshold	Long	Cleaning threshold.
block_threshold	String	Blackhole threshold.

Example Requests

None

Example Responses

Status code: 200

Request succeeded.

```
{
  "total" : 2,
  "ddosStatus" : [ {
    "floating_ip_id" : "xxxxxxxx-xxxx-4947-9fa8-dda843ae5d1d",
    "floating_ip_address" : "10.10.10.10",
    "status" : "normal",
    "product_type" : "CNAD",
    "clean_threshold" : 1000,
    "block_threshold" : ">=20Gbps"
  }, {
    "floating_ip_id" : "xxxxxxxx-xxxx-4d33-a2dc-2ace2d0283de",
    "floating_ip_address" : "11.11.11.11",
    "status" : "normal",
    "product_type" : "Anti-DDoS",
    "clean_threshold" : 120,
    "block_threshold" : ">=20Gbps"
  } ]
}
```

Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.2.10 Enabling Anti-DDoS

Function

This API is used to enable Anti-DDoS.

Calling Method

For details, see **Calling APIs**.

URI

POST /v1/{project_id}/antiddos/{floating_ip_id}

Table 4-51 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.
floating_ip_id	Yes	String	ID corresponding to the EIP of a user.

Request Parameters

Table 4-52 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Table 4-53 Request body parameters

Parameter	Mandatory	Type	Description
app_type_id	Yes	Long	Application type ID. The value is fixed to **0**.
cleaning_access_pos_id	Yes	Long	Position ID of access limit during cleaning. The options are as follows: **1**: 10 Mbit/s; **2**: 30 Mbit/s; **3**: 50 Mbit/s; **4**: 70 Mbit/s; **5**: 100 Mbit/s; **6**: 150 Mbit/s; **7**: 200 Mbit/s; **8**: 250 Mbit/s; **9**: 300 Mbit/s; **10**: 500 Mbit/s; **11**: 800 Mbit/s;**88**: 1,000 Mbit/s; **99**: default protection
enable_L7	Yes	Boolean	Whether to enable layer-7 protection. The value is fixed to **false**.
http_request_pos_id	Yes	Long	Segment ID of the HTTP request counts. The value is fixed to **1**.
traffic_pos_id	Yes	Long	Position ID of traffic. The options are as follows: **1**: 10 Mbit/s; **2**: 30 Mbit/s; **3**: 50 Mbit/s; **4**: 70 Mbit/s; **5**: 100 Mbit/s; **6**: 150 Mbit/s; **7**: 200 Mbit/s; **8**: 250 Mbit/s; **9**: 300 Mbit/s; **10**: 500 Mbit/s; **11**: 800 Mbit/s;**88**: 1,000 Mbit/s; **99**: default protection
antiddos_config_id	No	String	Protection level ID.

Response Parameters

Status code: 200

Table 4-54 Response body parameters

Parameter	Type	Description
error_code	String	Internal error code.
error_msg	String	Internal error description.
task_id	String	ID of a task. This ID can be used to query the status of the task. This field is reserved for future task audit extension and is not required currently.

Example Requests

Enable the Anti-DDoS policy for a specified EIP. Set the position ID of access limit during cleaning to 8, and set the position ID of traffic to 1.

```
POST https://{endpoint}/v1/{project_id}/antiddos/{floating_ip_id}

{
  "app_type_id" : 0,
  "cleaning_access_pos_id" : 8,
  "enable_L7" : false,
  "http_request_pos_id" : 1,
  "traffic_pos_id" : 1,
  "antiddos_config_name" : "test",
  "antiddos_config_id" : "1234567890"
}
```

Example Responses

Status code: 200

Request succeeded.

```
{
  "error_code" : "10000000",
  "error_msg" : "The task has been received and is being handled",
  "task_id" : "59385d2a-6266-4d3a-9122-a228c530f557"
}
```

Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.2.11 Querying Quotas

Function

Query quotas.

Calling Method

For details, see **Calling APIs**.

URI

GET /v1/{project_id}/antiddos/quotas

Table 4-55 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.

Request Parameters

Table 4-56 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-57 Response body parameters

Parameter	Type	Description
ddos_quota	**ddos_quota** object	Quota information.

Table 4-58 ddos_quota

Parameter	Type	Description
current	Integer	Quota used by the current user.
quota	Integer	Maximum quota.

Example Requests

None

Example Responses

Status code: 200

OK

```
{
  "ddos_quota" : {
    "current" : 3,
    "quota" : 350
  }
}
```

Status Codes

Status Code	Description
200	OK
401	Unauthorized
403	Forbidden
404	Not Found

Error Codes

See **Error Codes**.

4.2.12 Updating All User Log Settings

Function

Updating all user log settings.

Calling Method

For details, see **Calling APIs**.

URI

PUT /v1/{project_id}/antiddos/lts-config

Table 4-59 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.

Table 4-60 Query Parameters

Parameter	Mandatory	Type	Description
enterprise_project_id	Yes	String	Enterprise project ID.

Request Parameters

Table 4-61 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Table 4-62 Request body parameters

Parameter	Mandatory	Type	Description
enabled	No	Boolean	Whether to enable logging.
lts_id_info	No	**lts_id_info** object	Log information.

Table 4-63 lts_id_info

Parameter	Mandatory	Type	Description
lts_group_id	No	String	Log group ID.

Parameter	Mandatory	Type	Description
lts_attack_stream_id	No	String	Log stream ID.

Response Parameters

Status code: 200

OK

None

Example Requests

None

Example Responses

Status code: 200

OK

```
{
  "error_code" : "0",
  "error_msg" : "Success."
}
```

Status Codes

Status Code	Description
200	OK
401	Unauthorized
403	Forbidden
404	Not Found

Error Codes

See **Error Codes**.

4.2.13 Querying All Log Settings

Function

Query all log settings.

Calling Method

For details, see **Calling APIs**.

URI

GET /v1/{project_id}/antiddos/lts-config

Table 4-64 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.

Table 4-65 Query Parameters

Parameter	Mandatory	Type	Description
enterprise_project_id	Yes	String	Enterprise project ID.

Request Parameters

Table 4-66 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-67 Response body parameters

Parameter	Type	Description
enabled	Boolean	Whether to enable logging.
lts_id_info	**lts_id_info** object	Log information.

Table 4-68 lts_id_info

Parameter	Type	Description
lts_group_id	String	Log group ID.
lts_attack_stream_id	String	Log stream ID.

Example Requests

None

Example Responses

Status code: 200

Request succeeded.

```
{
  "enabled" : true,
  "lts_id_info" : {
    "lts_group_id" : "3f664a8c-53bd-4f77-922d-xxxxxxxxxxxx",
    "lts_attack_stream_id" : "ea9ffec3-6fe4-4884-9dd0-xxxxxxxxxxxx"
  }
}
```

Status Codes

Status Code	Description
200	Request succeeded.
401	Unauthorized
403	Forbidden
404	Not Found

Error Codes

See **Error Codes**.

4.3 Alarm Configuration Management

4.3.1 Querying Alarm Configuration

Function

You can query alarm configuration, such as whether a certain type of alarms will be received, and whether alarms are received through SMS messages or emails.

Calling Method

For details, see **Calling APIs**.

URI

GET /v2/{project_id}/warnalert/alertconfig/query

Table 4-69 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.

Request Parameters

Table 4-70 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Response Parameters

Status code: 200

Table 4-71 Response body parameters

Parameter	Type	Description
topic_urn	String	Unique ID of an alarm group.
display_name	String	Alarm group description.
warn_config	**warn_config** object	Alarm configuration information.

Table 4-72 warn_config

Parameter	Type	Description
antiDDoS	Boolean	DDoS attack.
back_doors	Boolean	Web shell.
high_privilege	Boolean	Excessive privileges assigned to a database process.
remote_login	Boolean	Alarms about remote logins.
send_frequency	Integer	**Range** • **0**: once a day • **1**: once every 30 minutes Mandatory for the HID.
waf	Boolean	Reserved field.
weak_password	Boolean	Weak password (system and database).

Example Requests

None

Example Responses

Status code: 200

Request succeeded.

```
{
  "warn_config" : {
    "antiDDoS" : false,
    "remote_login" : false,
    "weak_password" : false,
    "high_privilege" : false,
    "back_doors" : false,
    "waf" : false,
    "send_frequency" : 0
  },
  "topic_urn" : null,
  "display_name" : null
}
```

Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

4.3.2 Updating Alarm Configuration

Function

This API allows you to update alarm configuration, such as whether a certain type of alarms will be received, and whether alarms are received through SMS messages or emails.

Calling Method

For details, see **Calling APIs**.

URI

POST /v2/{project_id}/warnalert/alertconfig/update

Table 4-73 Path Parameters

Parameter	Mandatory	Type	Description
project_id	Yes	String	Project ID.

Request Parameters

Table 4-74 Request header parameters

Parameter	Mandatory	Type	Description
X-Auth-Token	Yes	String	User token. It can be obtained by calling the IAM API used to obtain a user token. The value of **X-Subject-Token** in the response header is the user token.
Content-Type	Yes	String	Content-Type request header.

Table 4-75 Request body parameters

Parameter	Mandatory	Type	Description
display_name	Yes	String	Alarm group description.
topic_urn	Yes	String	Unique ID of an alarm group.

Parameter	Mandatory	Type	Description
warn_config	Yes	WarnConfig object	Alarm configuration.

Table 4-76 WarnConfig

Parameter	Mandatory	Type	Description
antiDDoS	Yes	Boolean	DDoS attack.
back_doors	No	Boolean	Web shell.
high_privilege	No	Boolean	Excessive privileges assigned to a database process.
remote_login	No	Boolean	Alarms about remote logins.
send_frequency	No	Integer	**Range** • **0**: once a day • **1**: once every 30 minutes Mandatory for the HID.
waf	No	Boolean	Reserved field.
weak_password	No	Boolean	Weak password (system and database).

Response Parameters

Status code: 200

Table 4-77 Response body parameters

Parameter	Type	Description
error_code	String	Internal error code.
error_msg	String	Internal error description.

Example Requests

Set the SMN topic to **urn:smn:region-7:2d2d90a56a3243bdb909f6a24a27be8d:cnad-test-intl** and set the attack notification type to DDoS attacks.

```
POST https://{endpoint}/v2/{project_id}/warnalert/alertconfig/update

{
  "display_name" : "",
  "topic_urn" : "urn:smn:region-7:2d2d90a56a3243bdb909f6a24a27be8d:cnad-test-intl",
```



```
 "warn_config" : {
   "antiDDoS" : true,
   "back_doors" : false,
   "high_privilege" : false,
   "remote_login" : false,
   "send_frequency" : 1,
   "waf" : false,
   "weak_password" : false
 }
}
```

Example Responses

Status code: 200

Request succeeded.

```
{
  "error_code" : "10000000",
  "error_msg" : "Ok",
  "task_id" : ""
}
```

Status Codes

Status Code	Description
200	Request succeeded.

Error Codes

See **Error Codes**.

5 Examples

5.1 Example 1: Updating Defense Policy for an IP Address

Scenario

You can modify the defense policy configured for an IP address on the Anti-DDoS console or by calling the API.

Process:

1. Query the defense statuses of all IP addresses.
2. Query the optional Anti-DDoS defense policies.
3. Update the defense policy for an IP address.
4. Query the execution status of the defense policy updating task based on the task ID returned in **3**.

Involved APIs

The following APIs are required for updating the Anti-DDoS defense policy for an IP address:

- **Step 1** is used for querying the defense statuses of all IP addresses.
- **Step 2** is used for querying the optional Anti-DDoS defense policies.
- **Step 3** is used for updating the defense policy for an IP address.
- **Step 4** is used for querying the execution status of the defense policy updating task based on the task ID.

Procedure

Step 1 Query the defense statuses of all IP addresses.

- API information

  URI format: **GET /v1/{*project_id*}/antiddos**

  For details, see section "Querying the List of Defense Statuses of EIPs".

- Example request

  GET: https://*{endpoint}*/v1/1858a4e1f99d4454bd6a539d5477f5de/antiddos

  Obtain *{endpoint}* from .

  Body:

  ```
  {
  }
  ```

- Example response

  ```
  {
  "total": 1,
  "ddosStatus": [
  {
  "floating_ip_id": "18e6ace5-eb36-4196-a15e-1e000c24e026",
  "floating_ip_address": "139.9.116.167",
  "network_type": "EIP",
  "status": "normal",
  "blackhole_endtime": 0,
  "protect_type": "default",
  "traffic_threshold": 99,
  "http_threshold": 0
  }
  ]
  }
  ```

Step 2 Query the optional Anti-DDoS defense policies.

- API information

  URI format: **GET /v2/{*project_id*}/antiddos/query-config-list**

  For details, see section "Querying Optional Anti-DDoS Defense Policies".

- Example request

  GET: https://*{endpoint}*/v2/1858a4e1f99d4454bd6a539d5477f5de/antiddos/query-config-list

  Obtain *{endpoint}* from .

  Body:

  ```
  {
  }
  ```

- Example response

  ```
  {
  "traffic_limited_list": [
  {
  "traffic_pos_id": 1,
  "traffic_per_second": 10,
  "packet_per_second": 2000
  },
  {
  "traffic_pos_id": 2,
  "traffic_per_second": 30,
  "packet_per_second": 6000
  },
  {
  "traffic_pos_id": 3,
  "traffic_per_second": 50,
  "packet_per_second": 10000
  },
  {
  "traffic_pos_id": 4,
  "traffic_per_second": 70,
  "packet_per_second": 15000
  },
  {
  "traffic_pos_id": 5,
  "traffic_per_second": 100,
  ```



```
"packet_per_second": 20000
},
{
"traffic_pos_id": 6,
"traffic_per_second": 150,
"packet_per_second": 25000
},
{
"traffic_pos_id": 7,
"traffic_per_second": 200,
"packet_per_second": 35000
},
{
"traffic_pos_id": 8,
"traffic_per_second": 250,
"packet_per_second": 50000
},
{
"traffic_pos_id": 9,
"traffic_per_second": 300,
"packet_per_second": 70000
},
{
"traffic_pos_id": 88,
"traffic_per_second": 1000,
"packet_per_second": 300000
}
],
"http_limited_list": [
{
"http_request_pos_id": 1,
"http_packet_per_second": 100
},
{
"http_request_pos_id": 2,
"http_packet_per_second": 150
},
{
"http_request_pos_id": 3,
"http_packet_per_second": 240
},
{
"http_request_pos_id": 4,
"http_packet_per_second": 350
},
{
"http_request_pos_id": 5,
"http_packet_per_second": 480
},
{
"http_request_pos_id": 6,
"http_packet_per_second": 550
},
{
"http_request_pos_id": 7,
"http_packet_per_second": 700
},
{
"http_request_pos_id": 8,
"http_packet_per_second": 850
},
{
"http_request_pos_id": 9,
"http_packet_per_second": 1000
},
{
"http_request_pos_id": 10,
"http_packet_per_second": 1500
},
```



```
{
"http_request_pos_id": 11,
"http_packet_per_second": 2000
},
{
"http_request_pos_id": 12,
"http_packet_per_second": 3000
},
{
"http_request_pos_id": 13,
"http_packet_per_second": 5000
},
{
"http_request_pos_id": 14,
"http_packet_per_second": 10000
},
{
"http_request_pos_id": 15,
"http_packet_per_second": 20000
}
],
"connection_limited_list": [
{
"cleaning_access_pos_id": 1,
"new_connection_limited": 10,
"total_connection_limited": 30
},
{
"cleaning_access_pos_id": 2,
"new_connection_limited": 20,
"total_connection_limited": 100
},
{
"cleaning_access_pos_id": 3,
"new_connection_limited": 30,
"total_connection_limited": 200
},
{
"cleaning_access_pos_id": 4,
"new_connection_limited": 40,
"total_connection_limited": 250
},
{
"cleaning_access_pos_id": 5,
"new_connection_limited": 50,
"total_connection_limited": 300
},
{
"cleaning_access_pos_id": 6,
"new_connection_limited": 60,
"total_connection_limited": 500
},
{
"cleaning_access_pos_id": 7,
"new_connection_limited": 70,
"total_connection_limited": 600
},
{
"cleaning_access_pos_id": 8,
"new_connection_limited": 80,
"total_connection_limited": 700
}
],
"extend_ddos_config": []
}
```


Step 3 Update the defense policy for an IP address.

- API information

URI format: **PUT /v1/{*project_id*}/antiddos/{floating_ip_id}**

For details, see section "Updating Anti-DDoS Defense Policies".

- Example request

  PUT: https://*{endpoint}*/v1/1858a4e1f99d4454bd6a539d5477f5de/antiddos/18e6ace5-eb36-4196-a15e-1e000c24e026

  Obtain *{endpoint}* from .

  Body:

  ```
  {
  "app_type_id": 1,
  "cleaning_access_pos_id": 8,
  "enable_L7": false,
  "http_request_pos_id": 8,
  "traffic_pos_id": 8
  }
  ```

- Example response

  ```
  {
  "error_code": "10000000",
  "error_msg": "The task has been received and is being handled",
  "task_id": "59385d2a-6266-4d3a-9122-a228c530f557"
  }
  ```

Step 4 Query the execution status of the defense policy updating task based on the task ID returned in **Step 3**.

- API information

  URI format: **GET /v2/{*project_id*}/query-task-status**

  For details, see section "Querying Anti-DDoS Tasks".

- Example request

  GET: https://*{endpoint}*/v2/1858a4e1f99d4454bd6a539d5477f5de/query-task-status?task_id=59385d2a-6266-4d3a-9122-a228c530f557

  Obtain *{endpoint}* from .

  Body:

  ```
  {
  }
  ```

- Example response

  ```
  {
  "task_status": "success",
  "task_msg": ""
  }
  ```

----End

5.2 Example 2: Configuring the Default Protection Policy

Scenario

You can configure the default protection policy for newly purchased IP addresses on the Anti-DDoS console or by call the API.

Process:

1. Query the default protection settings for the newly purchased IP addresses.
2. Configure the default protection policy for newly purchased IP addresses.

Involved APIs

The following APIs are required for configuring default protection policy for newly purchased IP addresses.

- **Step 1** is used to query default protection settings for the newly purchased IP addresses.
- **Step 2** is used to configure default protection policy for newly purchased IP addresses.

Procedure

Step 1 Query default protection settings for the newly purchased IP addresses.

- API information

  URI format: **GET /v1/{*project_id*}/antiddos/default-config**

  For details, see section "Querying the Default Protection Policy Configured for the Newly Purchased Public IP Addresses".

- Example request

  GET: https://*{endpoint}*/v1/1858a4e1f99d4454bd6a539d5477f5de/antiddos/default/config

  Obtain *{endpoint}* from .

  Body:

  ```
  {
  }
  ```

- Example response

  ```
  {
  "app_type_id": 1,
  "cleaning_access_pos_id": 8,
  "enable_L7": false,
  "http_request_pos_id": 8,
  "traffic_pos_id": 8
  }
  ```

Step 2 Configure default protection policy for newly purchased IP addresses.

- API information

  URI format: **POST /v1/{*project_id*}/antiddos/default-config**

  For details, see section "Configuring the Default Protection Policy for Newly Purchased Public IP Addresses".

- Example request

  POST: https://*{endpoint}*/v1/1858a4e1f99d4454bd6a539d5477f5de/antiddos/default-config

  Obtain *{endpoint}* from .

  Body:

  ```
  {
  "app_type_id": 1,
  "cleaning_access_pos_id": 8,
  "enable_L7": false,
  "http_request_pos_id": 8,
  ```



```
"traffic_pos_id": 8
}
```

- Example response

```
{
}
```

----End

A Appendix

A.1 Status Code

- Normal

Returned Value	Description
200	The request is successfully processed.

- Abnormal

Status Code	Status	Description
400	Bad Request	The server fails to process the request.
401	Unauthorized	The requested page requires a username and a password.
403	Forbidden	Access to the requested page is denied.
404	Not Found	The server fails to find the requested page.
405	Method Not Allowed	Method specified in the request is not allowed.
406	Not Acceptable	Response generated by the server is not acceptable to the client.
407	Proxy Authentication Required	Proxy authentication is required before the request is processed.
408	Request Timeout	A timeout error occurs because the request is not processed within the specified waiting period of the server.

Status Code	Status	Description
409	Conflict	The request cannot be processed due to a conflict.
500	Internal Server Error	The request is not processed due to a server error.
501	Not Implemented	The request is not processed because the server does not support the requested function.
502	Bad Gateway	The request is not processed, and the server receives an invalid response from the upstream server.
503	Service Unavailable	The request is not processed due to a temporary system abnormality.
504	Gateway Timeout	A gateway timeout error occurs.

A.2 Error Codes

Status Code	Error Codes	Error Message	Description	Solution
200	Anti-DDoS.0	Succeeded	Succeeded	No need to deal with it
200	Anti-DDoS.10000000	The task has been received and is being handled	The task has been received and is being handled	No need to deal with it
400	Anti-DDoS.10000001	Enter a valid request message	Enter a valid request message	Check the parameters
400	Anti-DDoS.10001008	An incorrect task ID is used	An incorrect task ID is used	Check the parameters
400	Anti-DDoS.10001010	Invalid time	Invalid time	Check the parameters
401	Anti-DDoS.10000004	Public test service denied	Public test service denied	Apply for public test

Status Code	Error Codes	Error Message	Description	Solution
403	Anti-DDoS.10000002	Failed to authenticate the token in the request	Failed to authenticate the token in the request	Re apply for token
403	Anti-DDoS.10000009	The account is restricted	The account is restricted	Apply for authority
403	Anti-DDoS.10000010	The account is frozen	The account is frozen	Apply for unfreeze
403	Anti-DDoS.10000012	Unknown user type	Unknown user type	Apply for authority
403	Anti-DDoS.10000016	VPC access failed or EIP is not exist	VPC access failed or EIP is not exist	Contact to administrator
403	Anti-DDoS.10000030	You have not been authenticated. Perform real-name authentication first.	You have not been authenticated. Perform real-name authentication first.	Real name authentication
403	Anti-DDoS.10001009	The operation permission is restricted	The operation permission is restricted	Apply for authority
403	Anti-DDoS.11000001	Access to the database is rejected	Access to the database is rejected	Contact to administrator
500	Anti-DDoS.11000000	Internal system exception. Contact technical support engineers	Internal system exception. Contact technical support engineers	Contact to administrator

A.3 Obtaining a Project ID

Obtaining a Project ID by Calling an API

You can obtain the project ID by calling the IAM API used to query project information based on the specified criteria.

The API used to obtain a project ID is GET https://{Endpoint}/v3/projects. **{Endpoint}** is the IAM endpoint and can be obtained from the administrator. For details about API authentication, see **Authentication**.

In the following example, **id** indicates the project ID.

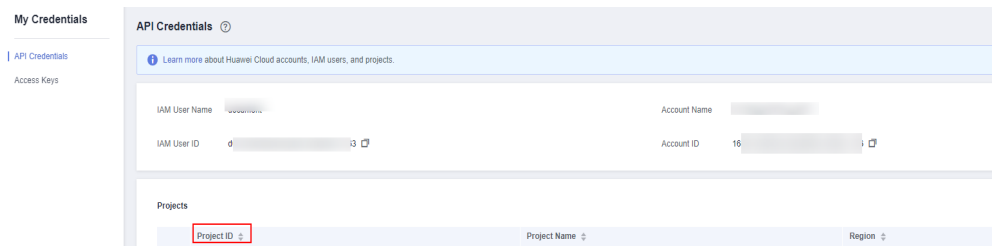

```
{
    "projects": [
        {
            "domain_id": "65382450e8f64ac0870cd180d14e684b",
            "is_domain": false,
            "parent_id": "65382450e8f64ac0870cd180d14e684b",
            "name": "xxxxxxxx",
            "description": "",
            "links": {
                "next": null,
                "previous": null,
                "self": "https://www.example.com/v3/projects/a4a5d4098fb4474fa22cd05f897d6b99"
            },
            "id": "a4a5d4098fb4474fa22cd05f897d6b99",
            "enabled": true
        }
    ],
    "links": {
        "next": null,
        "previous": null,
        "self": "https://www.example.com/v3/projects"
    }
}
```


Obtaining a Project ID from the Console

A project ID is required for some URLs when an API is called. To obtain a project ID, perform the following operations:

1. Log in to the management console.
2. Click the username and choose **My Credentials** from the drop-down list.
3. On the page, view the project ID in the project list.

Figure A-1 Viewing project IDs



B Out-of-Date APIs

B.1 Querying Optional Anti-DDoS Defense Policies

NOTE

This API is a historical API and may not be maintained in the future. Please use the API in section "Querying Optional Anti-DDoS Defense Policies".

Functions

This API allows you to query optional Anti-DDoS defense policies. Based on your service, you can select a policy for Anti-DDoS traffic cleaning.

URI

- URI format

  GET /v1/{project_id}/antiddos/query_config_list
- Parameter description

Parameter	Mandatory	Type	Description
project_id	Yes	String	User ID

Request

Example request

```
GET /v1/67641fe6886f43fcb78edbbf0ad0b99f/antiddos/query_config_list
```

Response

- Parameter description

Parameter	Mandatory	Type	Description
traffic_limited_list	Yes	List data structure	List of traffic limits
http_limited_list	Yes	List data structure	List of HTTP limits
connection_limited_list	Yes	List data structure	List of limits of numbers of connections

- Data structure description of **traffic_limited_list**

Parameter	Mandatory	Type	Description
traffic_pos_id	Yes	Integer	Position ID of traffic
traffic_per_second	Yes	Integer	Threshold of traffic per second (Mbit/s)
packet_per_second	Yes	Integer	Threshold of number of packets per second

- Data structure description of **http_limited_list**

Parameter	Mandatory	Type	Description
http_request_pos_id	Yes	Integer	Position ID of number of HTTP requests
http_packet_per_second	Yes	Integer	Threshold of number of HTTP requests per second

- Data structure description of **connection_limited_list**

Parameter	Mandatory	Type	Description
cleaning_access_pos_id	Yes	Integer	Position ID of access limit during cleaning
new_connection_limited	Yes	Integer	Number of new connections of a source IP address
total_connection_limited	Yes	Integer	Total number of connections of a source IP address

- Example response

```
{
  "traffic_limited_list": [
```



```
  {
    "traffic_pos_id": 1,
    "traffic_per_second": 10,
    "packet_per_second": 2000
  },
  {
    "traffic_pos_id": 2,
    "traffic_per_second": 30,
    "packet_per_second": 6000
  },
  {
    "traffic_pos_id": 3,
    "traffic_per_second": 50,
    "packet_per_second": 10000
  },
  {
    "traffic_pos_id": 4,
    "traffic_per_second": 70,
    "packet_per_second": 15000
  },
  {
    "traffic_pos_id": 5,
    "traffic_per_second": 100,
    "packet_per_second": 20000
  },
  {
    "traffic_pos_id": 6,
    "traffic_per_second": 150,
    "packet_per_second": 25000
  },
  {
    "traffic_pos_id": 7,
    "traffic_per_second": 200,
    "packet_per_second": 35000
  },
  {
    "traffic_pos_id": 8,
    "traffic_per_second": 250,
    "packet_per_second": 50000
  },
  {
    "traffic_pos_id": 9,
    "traffic_per_second": 300,
    "packet_per_second": 70000
  }
 ],
 "http_limited_list": [
  {
    "http_request_pos_id": 1,
    "http_packet_per_second": 100
  },
  {
    "http_request_pos_id": 2,
    "http_packet_per_second": 150
  },
  {
    "http_request_pos_id": 3,
    "http_packet_per_second": 240
  },
  {
    "http_request_pos_id": 4,
    "http_packet_per_second": 350
  },
  {
    "http_request_pos_id": 5,
    "http_packet_per_second": 480
  },
  {
    "http_request_pos_id": 6,
```



```
    "http_packet_per_second": 550
  },
  {
    "http_request_pos_id": 7,
    "http_packet_per_second": 700
  },
  {
    "http_request_pos_id": 8,
    "http_packet_per_second": 850
  },
  {
    "http_request_pos_id": 9,
    "http_packet_per_second": 1000
  },
  {
    "http_request_pos_id": 10,
    "http_packet_per_second": 1500
  },
  {
    "http_request_pos_id": 11,
    "http_packet_per_second": 2000
  },
  {
    "http_request_pos_id": 12,
    "http_packet_per_second": 3000
  },
  {
    "http_request_pos_id": 13,
    "http_packet_per_second": 5000
  },
  {
    "http_request_pos_id": 14,
    "http_packet_per_second": 10000
  },
  {
    "http_request_pos_id": 15,
    "http_packet_per_second": 20000
  }
],
"connection_limited_list": [
  {
    "cleaning_access_pos_id": 1,
    "new_connection_limited": 10,
    "total_connection_limited": 30
  },
  {
    "cleaning_access_pos_id": 2,
    "new_connection_limited": 20,
    "total_connection_limited": 100
  },
  {
    "cleaning_access_pos_id": 3,
    "new_connection_limited": 30,
    "total_connection_limited": 200
  },
  {
    "cleaning_access_pos_id": 4,
    "new_connection_limited": 40,
    "total_connection_limited": 250
  },
  {
    "cleaning_access_pos_id": 5,
    "new_connection_limited": 50,
    "total_connection_limited": 300
  },
  {
    "cleaning_access_pos_id": 6,
    "new_connection_limited": 60,
    "total_connection_limited": 500
```



```
  },
  {
    "cleaning_access_pos_id": 7,
    "new_connection_limited": 70,
    "total_connection_limited": 600
  },
  {
    "cleaning_access_pos_id": 8,
    "new_connection_limited": 80,
    "total_connection_limited": 700
  }
 ],
 "extend_ddos_config": [
  {
    "new_connection_limited": 80,
    "total_connection_limited": 700,
    "http_packet_per_second": 500000,
    "traffic_per_second": 1000,
    "packet_per_second": 200000,
    "setID": 33
  },
  {
    "new_connection_limited": 80,
    "total_connection_limited": 700,
    "http_packet_per_second": 500000,
    "traffic_per_second": 2000,
    "packet_per_second": 200000,
    "setID": 34
  },
  {
    "new_connection_limited": 80,
    "total_connection_limited": 700,
    "http_packet_per_second": 500000,
    "traffic_per_second": 5000,
    "packet_per_second": 400000,
    "setID": 35
  },
  {
    "new_connection_limited": 80,
    "total_connection_limited": 700,
    "http_packet_per_second": 0,
    "traffic_per_second": 0,
    "packet_per_second": 0,
    "setID": 36
  }
 ]
}
```


NOTE

The **extend_ddos_config** field displays information about Anti-DDoS defense policies set by users based on their needs.

Status Code

For details, see **Status Code**.

B.2 Querying Anti-DDoS Tasks

NOTE

This API is a historical API and may not be maintained in the future. Please use the API in section "Querying Anti-DDoS Tasks".

Functions

This API enables you to query the execution status of a specified Anti-DDoS configuration task.

URI

- URI format

  GET /v1/{project_id}/query_task_status

  NOTE

  You can use **?** and **&** behind the URI to add query conditions, as shown in the request example.

- Parameter description

Parameter	Mandatory	Type	Description
project_id	Yes	String	User ID

Request

- Parameter description

Parameter	Mandatory	Type	Description
task_id	Yes	String	Task ID (non-negative integer) character string

- Example request

```
GET /v1/67641fe6886f43fcb78edbbf0ad0b99f/query_task_status?
task_id=4a4fefe7-34a1-40e2-a87c-16932af3ac4a
```

Response

- Parameter description

Name	Type	Description
task_status	String	Status of a task, which can be one of the following: • success • failed • waiting • running • preprocess • ready
task_msg	String	Additional information about a task

- Example response


```
{
  "task_status": "running",
  "task_msg": ""
}
```

Status Code

For details, see **Status Code**.

