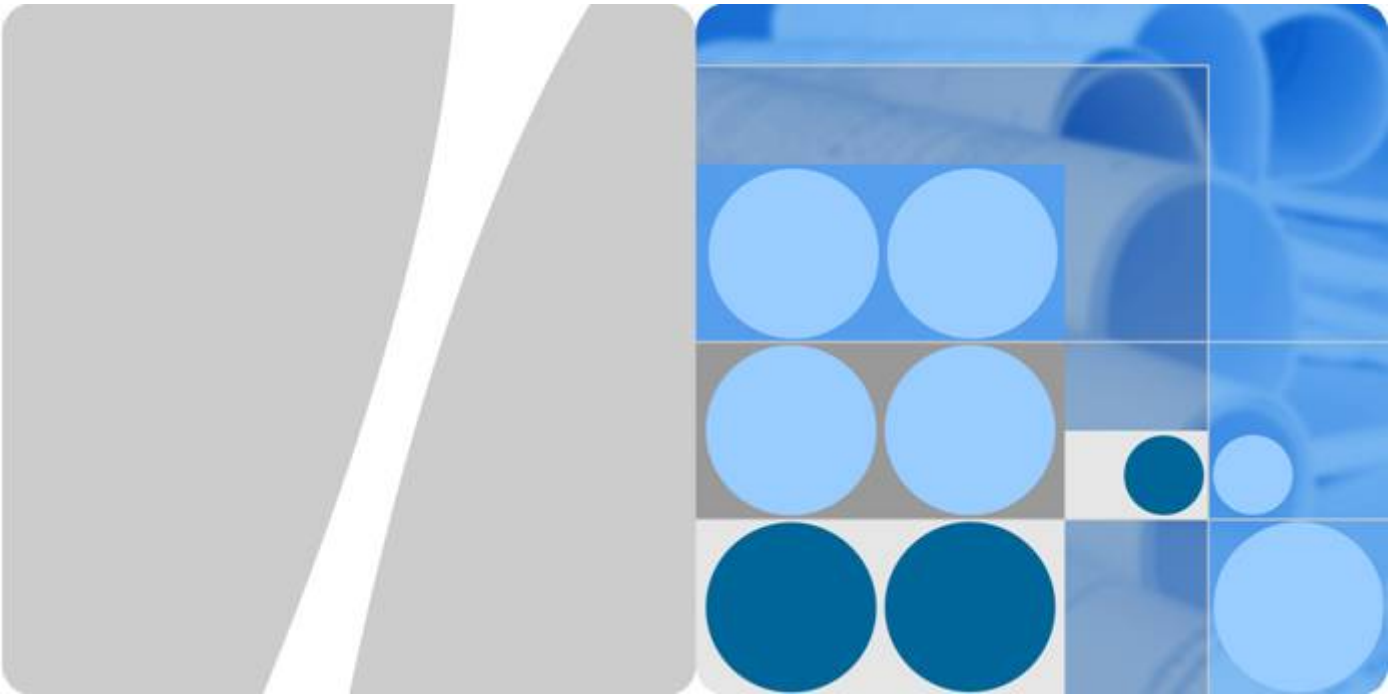


Dedicated Host

API Reference

Date **2021-11-12**

Contents

1 Before You Start

1.1 Overview

Welcome to *Dedicated Host API Reference*. Dedicated Host (DeH) provides dedicated physical hosts for you. You can create ECSs on a DeH to enhance isolation, security, and performance of your ECSs. When you migrate services to a DeH, you can continue to use your server software licenses used before the migration. That is, you can use the Bring Your Own License (BYOL) feature on the DeH to independently manage your ECSs.

This document describes how to use application programming interfaces (APIs) to perform operations on DeHs, such as creating, querying, deleting, and modifying DeHs. For details about all supported operations, see **API Overview**.

If you plan to access DeHs through an API, ensure that you are familiar with DeH concepts. For details, see "Service Overview" in the *Dedicated Host User Guide*.

1.2 API Calling

DeH supports Representational State Transfer (REST) APIs, allowing you to call APIs using HTTPS. For details about API calling, see **Calling APIs**.

1.3 Endpoints

An endpoint is the **request address** for calling an API. Endpoints vary depending on services and regions. For the endpoints of all services, see **Regions and Endpoints**.

1.4 Constraints

- The number of DeHs that you can create is determined by your quota. To view or increase the quota, see "Adjusting DeH Resource Quotas" in the *Dedicated Host User Guide*.
- For more constraints, see API description.

1.5 Concepts

- Domain

  A domain has full access permissions for all of its cloud services and resources. It can be used to reset user passwords and grant user permissions. The domain should not be used directly to perform routine management. For security purposes, create Identity and Access Management (IAM) users and grant them permissions for routine management.

- User

  An IAM user is created by an account in IAM to use cloud services. Each IAM user has its own identity credentials (password and access keys).

  API authentication requires information such as the domain name, username, and password.

- Region

  A region is a geographic area in which cloud resources are deployed. Availability zones (AZs) in the same region can communicate with each other over an intranet, while AZs in different regions are isolated from each other. Deploying cloud resources in different regions can better suit certain user requirements or comply with local laws or regulations.

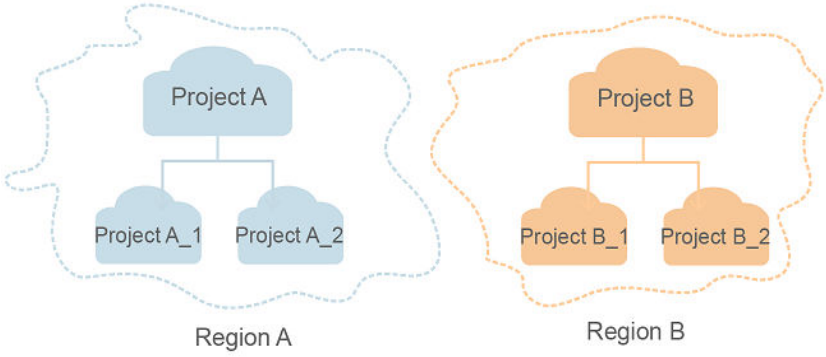
- AZ

  An AZ comprises of one or more physical data centers equipped with independent ventilation, fire, water, and electricity facilities. Computing, network, storage, and other resources in an AZ are logically divided into multiple clusters. AZs within a region are interconnected using high-speed optical fibers to allow you to build cross-AZ high-availability systems.

- Project

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

  Figure 1-1 Project isolation model



2 API Overview

You can use all DeH functions by making calls to the APIs provided by the DeH service, for example, allocating DeHs, querying the DeH list, viewing DeH details, and releasing a DeH.

Table 2-1 APIs

API	Description
Allocating DeHs	Allocate one or more DeHs and set required parameters, such as the flavor, AZ, and quantity. The number of allocatable DeHs depends on the DeH quota owned by the tenant.
Querying DeHs	Query the DeH list. You can add parameters such as **flavor**, **dedicated_host_id**, and **state** to the URI to filter the result.
Querying Details About a DeH	Query details about a DeH, such as its name, AZ, number of available vCPUs, and available memory size.
Querying ECSs on a DeH	Query information about the ECSs deployed on a DeH, such as the names, IDs, and status of the ECSs.
Modifying DeH Attributes	Change the name of a DeH and enable or disable the auto placement function. After auto placement is enabled, ECSs can be automatically scheduled to this DeH.
Releasing a DeH	Released a DeH that is no longer used.
Querying Available DeH Types	Query available DeH types in an AZ.
Querying API Versions	Query all available API versions and the versions of specified APIs.
DeH Tag Management	Add or delete tags for DeHs and query DeHs by tag.
Quota Configuration	Query the DeH quota of a tenant.

NOTE

For details about the APIs for creating ECSs on DeHs, see the *Elastic Cloud Server API Reference*.

3 Calling APIs

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.

Table 3-1 URI parameter description

Parameter	Description
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.

Table 3-2 HTTP methods

Method	Description
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://{{endpoint}}/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.

Table 3-3 Common request header fields

Parameter	Description	Mandatory	Example Value
Host	Specifies the server domain name and port number of the resources being requested. The value can be obtained from the URL of the service API. The value is in the format of *Hostname:Port number*. If the port number is not specified, the default port is used. The default port number for **https** is **443**.	No This field is mandatory for AK/SK authentication.	code.test.com or code.test.com:443
Content-Type	Specifies the type (or format) of the message body. The default value **application/json** is recommended. Other values of this field will be provided for specific APIs if any.	Yes	application/json
Content-Length	Specifies the length of the request body. The unit is byte.	No	3495
X-Project-Id	Specifies the project ID. Obtain the project ID by following the instructions in **Obtaining a Project ID**.	No	e9993fc787d94b6c886cbaa340f9c0f4
X-Auth-Token	Specifies the user token. It is a response to the API for **obtaining a user token** (This is the only API that does not require authentication). After the request is processed, the value of **X-Subject-Token** in the response header is the token value.	No This field is mandatory for token authentication.	The following is part of an example token: MIIPAgYJKoZIhvcNAQcCo...ggg1BBIINPXsidG9rZ

> **NOTE**
>
> In addition to supporting authentication using tokens, APIs support authentication using AK/SK, which uses SDKs to sign a request. During the signature, the **Authorization** (signature authentication) and **X-Sdk-Date** (time when a request is sent) headers are automatically added in the request.
>
> For more details, see "Authentication Using AK/SK" in **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://{{endpoint}}/v3/auth/tokens
Content-Type: application/json
```

(Optional) Request Body

This part is optional. 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. Replace *username*, *domainname*, ********* (login password), and *xxxxxxxxxxxxxxxxx* (project name) with the actual values. Obtain a project name 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://{{endpoint}}/v3/auth/tokens
Content-Type: application/json

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

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 a token.

Token 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. You can obtain a token by calling the **Obtaining User Token** API.

A cloud service can be deployed as either a project-level service or global service.

- For a project-level service, you need to obtain a project-level token. When you call the API, set **auth.scope** in the request body to **project**.
- For a global service, you need to obtain a global token. When you call the API, set **auth.scope** in the request body to **domain**.

IMS is a project-level service. When you call the API, set **auth.scope** in the request body to **project**.

```
{
    "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://{{endpoint}}/v3/auth/projects
Content-Type: application/json
X-Auth-Token: ABCDEFJ....
```

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 Codes**.

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**.

Figure 3-1 shows the response header fields for the API used to **obtain a user token**. The **x-subject-token** header field 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 is part of the response body for the API used to **obtain a user token**.

```
{
    "token": {
```

```
        "expires_at": "2019-02-13T06:52:13.855000Z",
        "methods": [
            "password"
        ],
        "catalog": [
            {
                "endpoints": [
                    {
                        "region_id": "az-01",
.....
```

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 Allocating DeHs

Function

This API is used to allocate one or more DeHs and set required parameters, such as the flavor, AZ, and quantity.

Constraints

The number of allocatable DeHs depends on the DeH quota owned by the tenant.

URI

POST /v1.0/{project_id}/dedicated-hosts

Table 4-1 describes the parameters.

Table 4-1 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.

Request

- Request parameters

  Table 4-2 Request parameters

Parameter	In	Type	Mandatory	Description
name	body	String	Yes	Specifies the DeH name.

Parameter	In	Type	Mandatory	Description
auto_placement	body	String	No	Specifies whether to allow an ECS to be placed on any available DeH if its DeH ID is not specified during its creation. The value can be **on** or **off**. The default value is **on**.
availability_zone	body	String	Yes	Specifies the AZ to which the DeH belongs.
host_type	body	String	Yes	Specifies the DeH type.
quantity	body	Integer	Yes	Specifies the number of allocatable DeHs.
tags	body	Array of objects	No	Specifies the DeH tags.

Table 4-3 resource_tag field description

Parameter	Type	Mandatory	Description
key	String	Yes	Specifies the tag key. ● It contains a maximum of 36 Unicode characters. ● The value cannot be empty. ● It cannot contain the following ASCII characters: =*<>|/, ● It can contain letters, digits, hyphens (-), and underscores (_).
value	String	Yes	Specifies the tag value. ● It contains a maximum of 43 Unicode characters. ● It cannot contain the following ASCII characters: =*<>|/, ● It can contain letters, digits, hyphens (-), and underscores (_).

- Example request

```
POST https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/dedicated-hosts
{
    "availability_zone": "dc1.az1",
    "name": "high performance servers1",
    "auto_placement": "off",
    "host_type": "h1",
```

```
    "quantity": 2,
    "tags": [
        {
            "key": "key1",
            "value": "value1"
        }
    ]
}
```

Response

- Response parameters

  Table 4-4 Response parameters

Parameter	In	Type	Description
dedicated_host_ids	body	Array of strings	Specifies a group of IDs of allocated DeHs. The tenant can create ECSs on these DeHs.

- Example response

```
{
    "dedicated_host_ids": ["xxxxxxx1","xxxxxxx2"]
}
```

Status Code

Table 4-5 Returned error codes

Error Code	Description
403 Forbidden	1. Insufficient quota. 2. The flavor is not supported.
404 FlavorNotFound	Invalid flavor.

For more status codes, see **Status Codes**.

4.2 Querying DeHs

Function

This API is used to query the DeH list.

URI

GET /v1.0/{project_id}/dedicated-hosts

Table 4-6 describes the parameters.

Table 4-6 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.

Request

- Request parameters

  You can add parameters **host_type**, **host_type_name**, **flavor**, **dedicated_host_id**, **state**, **tenant**, **availability_zone**, **name**, **limit**, **marker**, **tags**, **instance_uuid**, **released_at**, or **changes-since** to the URI to filter the search result,

  for example, **/v1.0/{project_id}/dedicated-hosts?host_type={host_type}&state={state}**.

  Table 4-7 Request parameters

Parameter	In	Type	Mandatory	Description
dedicated_host_id	query	String	No	Specifies the DeH ID.
name	query	String	No	Specifies the DeH name.
host_type	query	String	No	Specifies the DeH type.
host_type_name	query	String	No	Specifies the name of the DeH type.
flavor	query	String	No	Specifies the flavor ID.
state	query	String	No	Specifies the DeH status. The value can be **available**, **fault**, or **released**.
tenant	query	String	No	The value can be a tenant ID or **all**. Only the administrator can specify this parameter.
availability_zone	query	String	No	Specifies the AZ to which the DeH belongs.
limit	query	String	No	Specifies the number of records displayed per page.
marker	query	String	No	Specifies the ID of the last record on the previous page. If the **marker** value is invalid, status code 400 is returned.

Parameter	In	Type	Mandatory	Description
tags	query	String	No	Specifies the DeH tags.
instance_uuid	query	String	No	Specifies the ID of the ECS on the DeH.
released_at	query	String	No	Specifies the time when the DeH is released.
changes-since	query	String	No	Filters the response by date and timestamp when the DeH status changes. To help keep track of changes, this parameter may also display recently deleted DeHs. The format of the date and timestamp is ISO 8601: CCYY-MM-DDThh:mm:ss±hh:mm If the **hh:mm** value is included, the time zone is returned as the UTC offset, for example, **2015-08-27T09:49:58-05:00**. If you omit the time zone, the UTC time zone is assumed.

- Example request
  GET https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/dedicated-hosts?state=available

Response

- Response parameters

  Table 4-8 Response parameters

Parameter	In	Type	Description
dedicated_hosts	body	Array of objects For details, see **Table 5-1**.	Specifies the DeHs that meet the search criteria.
total	body	Integer	Specifies the quantity of DeHs that meet the search criteria.

- Example response

```
{
   "dedicated_hosts": [
      {
         "dedicated_host_id": "ab910cf0daebca90c4001",
         "name": "high performance servers1",
         "auto_placement": "off",
         "availability_zone": "az1",
         "host_properties": {
            "vcpus": 36,
```

```
            "cores": 12,
            "sockets": 2,
            "memory": 1073741824,
            "host_type": "h1",
            "host_type_name": "High performance",
            "available_instance_capacities": [
               {
                  "flavor": "h1.large"
               },
               {
                  "flavor": "h1.2large"
               },
               {
                  "flavor": "h1.4large"
               },
               {
                  "flavor": "h1.8large"
               }
            ]
         },
         "state": "available",
         "project_id": "9c53a566cb3443ab910cf0daebca90c4",
         "available_vcpus": 20,
         "available_memory": 1073201821,
         "instance_total": 2,
         "allocated_at": "2016-10-10T14:35:47Z",
         "released_at": null
         },
      {
         "dedicated_host_id": "ab910cf0daebca90c4002",
         "name": "high performance servers2",
         "auto_placement": "off",
         "availability_zone": "az1",
         "host_properties": {
            "vcpus": 36,
            "cores": 12,
            "sockets": 2,
            "host_type": "h1",
            "host_type_name": "High performance",
            "memory": 1073741824,
            "available_instance_capacities": [
               {
                  "flavor": "h1.large"
               },
               {
                  "flavor": "h1.2large"
               },
               {
                  "flavor": "h1.4large"
               },
               {
                  "flavor": "h1.8large"
               }
            ]
         },
         "state": "available",
         "project_id": "9c53a566cb3443ab910cf0daebca90c4",
         "available_vcpus": 20,
         "available_memory": 1073101821,
         "instance_total": 3,
         "allocated_at": "2016-10-10T14:35:47Z",
         "released_at": null
         },
      ...
   ],
   "total": 25
}
```

Status Code

See **Status Codes**.

4.3 Querying Details About a DeH

Function

This API is used to query details about a DeH.

URI

GET /v1.0/{project_id}/dedicated-hosts/{dedicated_host_id}

Table 4-9 describes the parameters.

Table 4-9 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.
dedicated_host_id	String	Yes	Specifies the DeH ID. You can obtain the DeH ID from the DeH console or using the **Querying DeHs** API.

Request

- Request parameters

  None

- Example request

```
GET https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/dedicated-hosts/ab910cf0daebca90c4001
```

Response

- Response parameters

  Table 4-10 Response parameters

Parameter	In	Type	Description
dedicated_host	body	Object For details, see **Table 5-1**.	Specifies the DeH object.

- Example response

```
{
    "dedicated_host": {
```

```
    "dedicated_host_id": "ab910cf0daebca90c4001",
    "name": "win_2008 servers",
    "auto_placement": "off",
    "availability_zone": "az1",
    "host_properties": {
        "vcpus": 36,
        "cores": 12,
        "sockets": 2,
        "memory": 1073741824,
        "host_type": "h1",
        "host_type_name": "High performance",
        "available_instance_capacities": [
            {
                "flavor": "h1.large"
            },
            {
                "flavor": "h1.2large"
            },
            {
                "flavor": "h1.4large"
            },
            {
                "flavor": "h1.8large"
            }
        ]
    },
    "state": "available",
    "project_id": "9c53a566cb3443ab910cf0daebca90c4",
    "available_vcpus": 20,
    "available_memory": 1073201821,
    "instance_total": 2,
    "allocated_at": "2016-10-10T14:35:47Z",
    "released_at": null,
    "instance_uuids": [
        "erf5th66cb3443ab912ff0daebca3456",
        "23457h66cb3443ab912ff0daebcaer45"
    ]
  }
}
```


Status Code

See **Status Codes**.

4.4 Querying ECSs on a DeH

Function

This API is used to query information about deployed ECSs on a DeH.

URI

GET /v1.0/{project_id}/dedicated-hosts/{dedicated_host_id}/servers

Table 4-11 describes the parameters.

Table 4-11 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.

Parameter	Type	Mandatory	Description
dedicated_host_id	String	Yes	Specifies the DeH ID. You can obtain the DeH ID from the DeH console or using the **Querying DeHs** API.

Request

- Request parameters

  Table 4-12 Request parameters

Parameter	In	Type	Mandatory	Description
limit	query	String	No	Specifies the number of records displayed per page.
marker	query	String	No	Specifies the ID of the last record on the previous page. If the **marker** value is invalid, status code 400 is returned.

- Example request

```
GET https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/dedicated-hosts/ab910cf0daebca90c4001/servers
```

Response

- Response parameters

  Table 4-13 Response parameters

Parameter	In	Type	Description
servers	body	Array of objects	Specifies the server object.

  Table 4-14 server field description

Parameter	Type	Description
addresses	Object (string:array)	Specifies the network attribute of the ECS. For details, see the **addresses** field description.

Parameter	Type	Description
created	String	Specifies the time when the ECS was created.
flavor	Object (string:string)	Specifies the ECS flavor.
id	String	Specifies the ECS ID in UUID format.
name	String	Specifies the ECS name.
status	String	Specifies the ECS status. Options: **ACTIVE**, **BUILD**, **DELETED**, **ERROR**, **HARD_REBOOT**, **MIGRATING**, **PASSWORD**, **PAUSED**, **REBOOT**, **REBUILD**, **RESIZE**, **REVERT_RESIZE**, **SHUTOFF**, **SHELVED**, **SHELVED_OFFLOADED**, **SOFT_DELETED**, **SUSPENDED**, and **VERIFY_RESIZE**
tenant_id	String	Specifies the ECS tenant ID in UUID format.
updated	String	Specifies the time when the ECS was updated last time.
user_id	String	Specifies the ID of the user who has created the ECS. The value is in UUID format.
task_state	String	Specifies the ECS task status.
image	Object (string:string)	Specifies the ECS image.
metadata	Object (string:string)	Specifies the ECS metadata.

- Example response

```
{
   "servers": [
      {
         "addresses": {
            "68269e6e-4a27-441b-8029-35373ad50bd9": [
               {
                  "addr": "192.168.0.3",
                  "version": 4
               }
            ]
         },
         "created": "2012-09-07T16:56:37Z",
         "flavor": {
            "id": "1"
         },
         "id": "05184ba3-00ba-4fbc-b7a2-03b62b884931",
         "metadata": {
            "os_type": "Linux"
         },
         "name": "new-server-test",
         "status": "ACTIVE",
         "tenant_id": "openstack",
```

```
            "updated": "2012-09-07T16:56:37Z",
            "user_id": "fake",
            "task_state": "",
            "image": {
                "id": "1ce5800a-e487-4c1b-b264-3353a39e2b4b"
            }
        }
    ]
}
```


Status Code

See **Status Codes**.

4.5 Modifying DeH Attributes

Function

This API is used to modify the **auto_placement** and **name** attributes of a DeH.

URI

PUT /v1.0/{project_id}/dedicated-hosts/{dedicated_host_id}

Table 4-15 describes the parameters.

Table 4-15 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.
dedicated_host_id	String	Yes	Specifies the DeH ID. You can obtain the DeH ID from the DeH console or using the **Querying DeHs** API.

Request

- Request parameters

Table 4-16 Request parameters

Parameter	In	Type	Mandatory	Description
auto_placement	in	String	No	Specifies whether to allow an ECS to be placed on any available DeH if its DeH ID is not specified during its creation. The value can be **on** or **off**.
name	in	String	No	Specifies the DeH name.

- Example request

```
PUT https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/dedicated-hosts/74259164-e63a-4ad9-9c77-a1bd2c9aa187
{
    "dedicated_host": {
        "auto_placement": "off",
        "name": "DeH_vm3"
    }
}
```

Response

- Response parameters

  None

- Example response

```
Http Response Code: 204
```

Status Code

See **Status Codes**.

4.6 Releasing a DeH

Function

This API is used to release a DeH.

Constraints

A DeH accommodating ECSs cannot be released.

URI

DELETE /v1.0/{project_id}/dedicated-hosts/{dedicated_host_id}

Table 4-17 describes the parameters.

Table 4-17 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.
dedicated_host_id	String	Yes	Specifies the DeH ID. You can obtain the DeH ID from the DeH console or using the **Querying DeHs** API.

Request

- Request parameters

  None

- Example request

```
DELETE https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/dedicated-hosts/74259164-e63a-4ad9-9c77-a1bd2c9aa187
```

Response

- Response parameters

  None

- Example response

```
Http Response Code: 204
```

Status Code

Table 4-18 Returned error codes

Error Code	Description
409 Conflict	A DeH accommodating ECSs cannot be released.

For more status codes, see **Status Codes**.

4.7 Querying Available DeH Types

Function

This API is used to query available DeH types in an AZ.

URI

Get /v1.0/{project_id}/availability-zone/{availability_zone}/dedicated-host-types

Table 4-19 describes the parameters.

Table 4-19 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.
availability_zone	String	Yes	Specifies the AZ.

Request

- Request parameters

  None

- Example request

```
GET https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/availability-zone/az1/dedicated-host-types
```

Response

- Response parameters

  Table 4-20 Response parameters

Parameter	In	Type	Description
dedicated_host_types	body	Array of objects	Specifies the available DeH types.
host_type	body	String	Specifies the DeH type.
host_type_name	body	String	Specifies the name of the DeH type.

- Example response

```
{
  "dedicated_host_types": [
    {
      "host_type": "General",
      "host_type_name": "General Computing"
    },
    {
      "host_type": "m1",
      "host_type_name": "Memory-optimized"
    },
    {
      "host_type": "h2",
      "host_type_name": "High performance"
    },
    {
      "host_type": "d1",
      "host_type_name": "Disk intensive"
    }
  ]
}
```

Status Code

See **Status Codes**.

4.8 API Version Query

4.8.1 Querying API Versions

Function

This API is used to query all API versions available to the DeH service.

URI

GET /

Request

- Request parameters

  None

- Example request

```
GET /
```

Response

- Response parameters

Table 4-21 Response parameters

Parameter	Type	Description
versions	Array of objects	Specifies the API versions.

Table 4-22 versions field description

Parameter	Type	Description
id	String	Specifies the ID of the API version.
links	Array of objects	Specifies the URL of the API version.
min_version	String	Specifies the microversion. If the APIs of this version support micro-versions, set this parameter to the supported minimum micro-version. If the microversion is not supported, leave this parameter blank.

Parameter	Type	Description
status	String	Specifies the API version status. • **CURRENT**: indicates a primary version. • **SUPPORTED**: indicates an earlier version which is still supported. • **DEPRECATED**: indicates a deprecated version which may be deleted later.
updated	String	Specifies the API version update time, which must be UTC time.
version	String	If the APIs of this version support micro-versions, set this parameter to the maximum micro-version supported. If not, leave this parameter blank.

Table 4-23 links field description

Parameter	Type	Description
href	String	Specifies the URL of the API version.
rel	String	Specifies the API URL dependency.

- Example response

```
{
   "versions": [
      {
         "id": "v1.0",
         "links": [
            {
               "href": "https//deh.xxx.com/v1.0/",
               "rel": "self"
            }
         ],
         "min_version": "",
         "status": "SUPPORTED",
         "updated": "2016-12-01T11:33:21Z",
         "version": ""
      }
   ]
}
```

Status Code

See **Status Codes**.

4.8.2 Querying an API Version

Function

This API is used to query a specified API version.

URI

GET /{api_version}

Table 4-24 describes the parameters.

Table 4-24 Parameters description

Parameter	Mandatory	Description
api_version	Yes	Specifies the API version, for example, v1.0.

Request

- Request parameters

  None

- Example request

```
GET /v1.0
```

Response

- Response parameters

Table 4-25 Response parameters

Parameter	Type	Description
version	Object	Specifies information about a specified API version.

Table 4-26 version field description

Parameter	Type	Description
id	String	Specifies the ID of the API version.
links	Array of objects	Specifies the URL of the API version.
min_version	String	Specifies the microversion. If the APIs of this version support micro-versions, set this parameter to the supported minimum micro-version. If the microversion is not supported, leave this parameter blank.

Parameter	Type	Description
status	String	Specifies the API version status. • **CURRENT**: indicates a primary version. • **SUPPORTED**: indicates an earlier version which is still supported. • **DEPRECATED**: indicates a deprecated version which may be deleted later.
updated	String	Specifies the API version update time.
version	String	If the APIs of this version support micro-versions, set this parameter to the maximum micro-version supported. If not, leave this parameter blank.

Table 4-27 links field description

Parameter	Type	Description
href	String	Specifies the URL of the API version.
rel	String	Specifies the API URL dependency.

- Example response

```
{
   "version": {
      "id": "v1.0",
      "links": [
         {
            "href": "https//deh.xxx.com/v1.0/",
            "rel": "self"
         }
      ],
      "min_version": "",
      "status": "SUPPORTED",
      "updated": "2016-12-01T11:33:21Z",
      "version": ""
   }
}
```

Status Code

See **Status Codes**.

4.9 DeH Tag Management

4.9.1 Adding Tags to a DeH in Batches

Function

- This API is used to add tags to a specified DeH in batches.
- Tag Management Service (TMS) uses this API to batch add tags to a DeH.

Constraint

- A DeH allows a maximum of 10 tags.
- This API is idempotent.

  During tag creation, if a tag exists (both the key and value are the same as those of an existing tag), the tag is successfully processed by default.
- A new tag will overwrite the original one if their keys are the same and values are different.

URI

POST /v1.0/{project_id}/dedicated-host-tags/{dedicated_host_id}/tags/action

Table 4-28 describes the parameters.

Table 4-28 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.
dedicated_host_id	String	Yes	Specifies the DeH ID. You can obtain the DeH ID from the DeH console or using the **Querying DeHs** API.

Request

- Request parameters

  Table 4-29 Request parameters

Parameter	Type	Mandatory	Description
tags	Array of objects	Yes	Specifies the tag list.

Parameter	Type	Mandatory	Description
action	String	Yes	Specifies the operation. Only lowercase letters are supported. For example, **create** indicates the creation operation.

Table 4-30 resource_tag field description

Parameter	Type	Mandatory	Description
key	String	Yes	Specifies the tag key. • It contains a maximum of 36 Unicode characters. • The value cannot be empty. • It cannot contain the following ASCII characters: =*<>|/, • It can contain letters, digits, hyphens (-), and underscores (_).
value	String	Yes	Specifies the tag value. • It contains a maximum of 43 Unicode characters. • It cannot contain the following ASCII characters: =*<>|/, • It can contain letters, digits, hyphens (-), and underscores (_).

- Example request

```
POST https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/dedicated-host-tags/74259164-e63a-4ad9-9c77-a1bd2c9aa187/tags/action
{
   "action": "create",
   "tags": [
      {
         "key": "key1",
         "value": "value1"
      },
      {
         "key": "key2",
         "value": "value2"
      }
   ]
}
```

Response

N/A

Status Code

See **Status Codes**.

4.9.2 Deleting Tags from a DeH in Batches

Function

- This API is used to delete tags from a specified DeH in batches.
- Tag Management Service (TMS) uses this API to batch delete tags from a DeH.

Constraint

A DeH allows a maximum of 10 tags.

URI

POST /v1.0/{project_id}/dedicated-host-tags/{dedicated_host_id}/tags/action

Table 4-31 describes the parameters.

Table 4-31 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.
dedicated_host_id	String	Yes	Specifies the DeH ID. You can obtain the DeH ID from the DeH console or using the **Querying DeHs** API.

Request

- Request parameters

Table 4-32 Request parameters

Parameter	Type	Mandatory	Description
tags	Array of objects	Yes	Specifies the tag list.

Parameter	Type	Mandatory	Description
action	String	Yes	Specifies the operation. Only lowercase letters are supported. For example, **delete** indicates the deletion operation.

Table 4-33 resource_tag field description

Parameter	Type	Mandatory	Description
key	String	Yes	Specifies the tag key. • It contains a maximum of 36 Unicode characters. • The value cannot be empty. • It cannot contain the following ASCII characters: =*<>|/, • It can contain letters, digits, hyphens (-), and underscores (_).
value	String	No	Specifies the tag value. • It contains a maximum of 43 Unicode characters. • It cannot contain the following ASCII characters: =*<>|/, • It can contain letters, digits, hyphens (-), and underscores (_).

- Example request

```
POST https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/dedicated-host-tags/74259164-e63a-4ad9-9c77-a1bd2c9aa187/tags/action
{
   "action": "delete",
   "tags": [
      {
         "key": "key1",
         "value": "value1"
      },
      {
         "key": "key2",
         "value": "value2"
      }
   ]
}
```

Response

N/A

Status Code

See **Status Codes**.

4.9.3 Querying Tags of a DeH

Function

- This API is used to query tags of a DeH.
- Tag Management Service (TMS) uses this API to query all tags of a DeH.

URI

GET /v1.0/{project_id}/dedicated-host-tags/{dedicated_host_id}/tags

Table 4-34 describes the parameters.

Table 4-34 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.
dedicated_host_id	String	Yes	Specifies the DeH ID. You can obtain the DeH ID from the DeH console or using the **Querying DeHs** API.

Request

- Request parameters

  None

- Example request

```
GET https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/dedicated-host-tags/74259164-e63a-4ad9-9c77-a1bd2c9aa187/tags
```

Response

- Response parameters

Table 4-35 Response parameters

Parameter	Type	Description
tags	Array of objects	Specifies the list of tags.

Table 4-36 resource_tag field description

Parameter	Type	Description
key	String	Specifies the tag key.
value	String	Specifies the tag value.

- Example response

```
{
   "tags": [
      {
         "key": "key1",
         "value": "value1"
      },
      {
         "key": "key2",
         "value": "value2"
      }
   ]
}
```

Status Code

See **Status Codes**.

4.9.4 Querying DeHs by Tag

Function

- This API is used to filter DeHs by tag and return the list of all tags of a DeH.
- Tag Management Service (TMS) uses this API to filter the DeHs.

URI

POST /v1.0/{project_id}/dedicated-host-tags/resource_instances/action

Table 4-37 describes the parameters.

Table 4-37 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.

Request

- Request parameters

Table 4-38 Request parameters

Parameter	Type	Mandatory	Description
tags	Array of objects	No	Displays all DeHs with specified tags. For more information, see **Table 4-39**. • A maximum of 10 keys can be included. Each key can have a maximum of 10 values. • The structure body must be included. • The tag key cannot be left blank or set to an empty string. • A key must be unique. • Values of the same key must be unique.
not_tags	Array of objects	No	Displays the DeHs with none of specified tags. • A maximum of 10 keys can be included. Each key can have a maximum of 10 values. • The structure body must be included. • The tag key cannot be left blank or set to an empty string. • Keys must be unique. • Values of the same key must be unique.
limit	String	No	Limits the maximum number of queried DeHs. The value cannot be a negative number. The maximum value is 1000. • If the **action** value is **count**, this parameter is invalid. • If the **action** value is **filter**, the default value is **1000**.

Parameter	Type	Mandatory	Description
offset	String	No	Specifies the index position. The query starts from the next piece of data indexed by this parameter. The value must be a non-negative number. You do not need to specify this parameter when querying resources on the first page. When you query resources on subsequent pages, set the value of **offset** to the location returned in the response body for the previous query. ● If the **action** value is **count**, this parameter is invalid. ● If the **action** value is **filter**, the default value is **0**.
action	String	Yes	Specifies the operation, which can be **filter** or **count**. ● **filter**: Filters DeHs by tag and lists DeHs that meet the search criteria. Listed DeHs are queried by page. ● **count**: Searches for DeHs by tag and returns the number of DeHs that meet the search criteria.
tags_any	Array of objects	No	Includes any of the specified tags. ● This field contains a maximum of 10 tag keys and each tag key has a maximum of 10 tag values. The tag value corresponding to each tag key can be an empty array but the structure cannot be missing. ● Each key must be unique, and cannot contain duplicate values. ● The response returns resources containing the tags in this list. Keys in this list are in an OR relationship and values in each key-value structure are also in an OR relationship. ● If no tag filtering condition is specified, full data is returned.

Parameter	Type	Mandatory	Description
not_tags_any	Array of objects	No	Excludes any of the specified tags. • This field contains a maximum of 10 tag keys and each tag key has a maximum of 10 tag values. The tag value corresponding to each tag key can be an empty array but the structure cannot be missing. • Each key must be unique, and cannot contain duplicate values. • The response returns resources containing no tags in this list. Keys in this list are in an OR relationship and values in each key-value structure are also in an OR relationship. • If no tag filtering condition is specified, full data is returned.
matches	Array of objects	No	Specifies the search field, which is used to search for DeHs by condition. Currently, only **resource_name** can be used for search. For more information, see **Table 4-40**.

Table 4-39 tag field description

Parameter	Type	Mandatory	Description
key	String	Yes	Specifies the tag key. • It contains a maximum of 127 Unicode characters. • This field cannot be left blank.

Parameter	Type	Mandatory	Description
values	Array of strings	No	Specifies the tag values. • Each tag contains a maximum of 10 values. • Values of the same tag must be unique. • Each value can contain a maximum of 255 Unicode characters. • If this parameter is not specified, any value can be used. • The resources containing one or more values listed in **values** will be found and displayed.

Table 4-40 match field description

Parameter	Type	Mandatory	Description
key	String	Yes	Specifies the key parameter to be matched. • The key must be unique, and the value is used for matching. • The **key** field is a fixed dictionary value. • This field cannot be left blank. **NOTE** The parameter value can only be **resource_name**, which is the DeH name.
value	String	Yes	Specifies the tag value. • Each value can contain a maximum of 255 Unicode characters. • This field cannot be left blank.

- Example request

```
POST https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/dedicated-host-tags/
resource_instances/action
{
   "offset": "0",
   "limit": "100",
   "action": "filter",
```

```
  "matches": [
    {
      "key": "resource_name",
      "value": "resource1"
    }
  ],
  "tags": [
    {
      "key": "key1",
      "values": ["value1"]
    }
  ]
}
```

Response

- Response parameters

Table 4-41 Response parameters

Parameter	Type	Description
resources	Array of objects	Specifies the returned DeH list. For details, see **Table 4-42**.
total_count	Integer	Specifies the total number of resources.

Table 4-42 Description of the **resource** field

Parameter	Type	Description
resource_id	String	Specifies the DeH ID.
resouce_detail	String	Specifies the DeH details. This field is used for future extension and is left empty by default.
tags	Array of objects	Specifies the tag list.
resource_name	String	Specifies the resource name.

Table 4-43 resource_tag field description

Parameter	Type	Description
key	String	Specifies the tag key. • It contains a maximum of 36 Unicode characters. • This field cannot be left blank. • It cannot contain the following ASCII characters: =*<>|/,

Parameter	Type	Description
value	String	Specifies the tag value. • Each value contains a maximum of 43 Unicode characters. • This field can be left blank. • It cannot contain the following ASCII characters: =*<>|/,

- Example response

  Response body when **action** is set to **filter**

```
{
   "resources": [
      {
         "resource_detail": null,
         "resource_id": "cdfs_cefs_wesas_12_dsad",
         "resource_name": "resource1",
         "tags": [
            {
               "key": "key1",
               "value": "value1"
            }
         ]
      }
   ],
   "total_count": 1
}
```

  Response body when **action** is set to **count**

```
{
   "total_count": 100
}
```

Status Code

See **Status Codes**.

4.10 Quota Configuration

4.10.1 Querying the DeH Quota of a Tenant

Function

This API is used to query the DeH quota of a tenant.

URI

GET /v1.0/{project_id}/quota-sets/{tenant_id}

Table 4-44 describes the parameters.

Table 4-44 Parameters description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Specifies the project ID.
tenant_id	String	Yes	Specifies the tenant ID. You can obtain the DeH ID from the DeH console or using the **Querying DeHs** API.

Request

- Request parameters

  You can add the **resource** parameter to the URI. For example:

  /v1.0/{project_id}/quota-sets/{tenant_id}?resource={resource}

  Table 4-45 Request parameters

Parameter	In	Type	Mandatory	Description
resource	query	String	No	Specifies the resource type.

- Example request

```
GET https://{Endpoint}/v1.0/9c53a566cb3443ab910cf0daebca90c4/quota-sets/
45df5566cb3443ab910cf0daebcapoi8
```

Response

- Response parameters

  Table 4-46 Response parameters

Parameter	Type	Description
quota_set	Array of objects	Specifies the quota set of a DeH.
resource	String	Specifies the resource type.
hard_limit	Integer	Specifies the quota limit. **-1** indicates that the resource quota is not limited.
used	Integer	Specifies the used amount of the quota.

- Example response

```
{
   "quota_set": [
      {
         "resource": "c1",
         "hard_limit": 5,
         "used": 2
      },
      {
         "resource": "m1",
         "hard_limit": 5,
         "used": 0
      },
      {
         "resource": "h1",
         "hard_limit": 5,
         "used": 2
      },
      {
         "resource": "d1",
         "hard_limit": 5,
         "used": 2
      }
   ]
}
```

Status Code

See **Status Codes**.

5 Public Parameters

5.1 Object Models

Objects

DeH management includes querying the DeH list, viewing DeH details, modifying DeH attributes, allocating a DeH, and releasing a DeH.

Object Models

Table 5-1 dedicated_host

Parameter	Type	CRUD	Default Value	Constraint	Description
dedicated_host_id	String	R	N/A	N/A	Specifies the DeH ID.
name	String	CUR	N/A	N/A	Specifies the DeH name. The name can contain a maximum of 255 characters and cannot start or end with spaces.
auto_placement	String	CUR	on	The value can be **on** or **off**.	Specifies whether to allow an ECS to be placed on any available DeH if its DeH ID is not specified during its creation.
availability_zone	String	CR	N/A	N/A	Specifies the AZ to which the DeH belongs.

Parameter	Type	CRUD	Default Value	Constraint	Description
project_id	String	CR	N/A	N/A	Specifies the tenant who owns the DeH.
host_properties	Dict For details, see **Table 5-2**.	R	N/A	N/A	Specifies the DeH properties.
state	String	R	N/A	The value can be **available**, **fault**, or **released**.	Specifies the DeH status.
available_vcpus	Int	R	N/A	N/A	Specifies the number of available vCPUs for the DeH.
available_memory	Int	R	N/A	N/A	Specifies the available memory size of the DeH.
allocated_at	String	R	N/A	N/A	Specifies the time when the DeH is allocated.
released_at	String	R	N/A	N/A	Specifies the time when the DeH is released.
instance_total	Int	R	N/A	N/A	Specifies the total number of ECSs on the DeH.
instance_uuids	List <String>	R	N/A	N/A	Specifies the UUIDs of the ECSs running on the DeH. This parameter is not displayed on the interface for **querying DeHs**.
tags	Dict(str: str)	R	N/A	N/A	Specifies the DeH tags.
sys_tags	Dict(str: str)	R	N/A	N/A	Specifies the DeH system tags.

Table 5-2 host_property

Parameter	Type	CRUD	Default Value	Constraint	Description
host_type	String	R	N/A	N/A	Specifies the DeH type.
host_type_name	String	R	N/A	N/A	Specifies the name of the DeH type.
vcpus	Int	R	N/A	N/A	Specifies the number of vCPUs on the DeH.
cores	Int	R	N/A	N/A	Specifies the number of physical cores on the DeH.
sockets	Int	R	N/A	N/A	Specifies the number of physical sockets on the DeH.
memory	Int	R	N/A	N/A	Specifies the size of physical memory on the DeH.
available_instance_capacities	List For details, see **Table 5-3**.	R	N/A	N/A	Specifies the flavors of ECSs placed on the DeH.

Table 5-3 available_instance_capacity

Parameter	Type	CRUD	Default Value	Constraint	Description
flavor	String	R	N/A	N/A	Specifies the specifications of ECSs that can be created.

5.2 Status Codes

- Normal

Returned Value	Description
200 OK	The server has successfully processed the request.
201 Created	The request is successful and a resource is created on the server.
202 Accepted	The request has been accepted, but the processing has been delayed.
204 No Content	The server has processed the request but did not return any content.

- Abnormal

Returned Value	Description
400 Bad Request	The server failed to process the request.
401 Unauthorized	You must enter a username and password to access the requested page.
403 Forbidden	You are forbidden to access the requested page.
404 Not Found	The server could not find the requested page.
405 Method Not Allowed	You are not allowed to use the method specified in the request.
406 Not Acceptable	The response generated by the server could not be accepted by the client.
407 Proxy Authentication Required	You must use the proxy server for authentication so that the request can be processed.
408 Request Timeout	The request timed out.

Returned Value	Description
409 Conflict	The request could not be processed due to a conflict.
500 Internal Server Error	Failed to complete the request because of an internal service error.
501 Not Implemented	Failed to complete the request because the server does not support the requested function.
502 Bad Gateway	Failed to complete the request because the server has received an invalid response.
503 Service Unavailable	Failed to complete the request because the service is unavailable.
504 Gateway Timeout	A gateway timeout error occurred.

5.3 Obtaining a Project ID

Scenarios

A project ID is required for some URLs when an API is called. Therefore, you need to obtain a project ID in advance. Two methods are available:

- **Obtain the Project ID by Calling an API**
- **Obtain the Project ID from the Console**

Obtain the 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**.

The following is an example response. The value of **id** is the project ID.

```
{
    "projects": [
        {
            "domain_id": "65382450e8f64ac0870cd180d14e684b",
            "is_domain": false,
            "parent_id": "65382450e8f64ac0870cd180d14e684b",
            "name": "project_name",
            "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"
    }
}
```

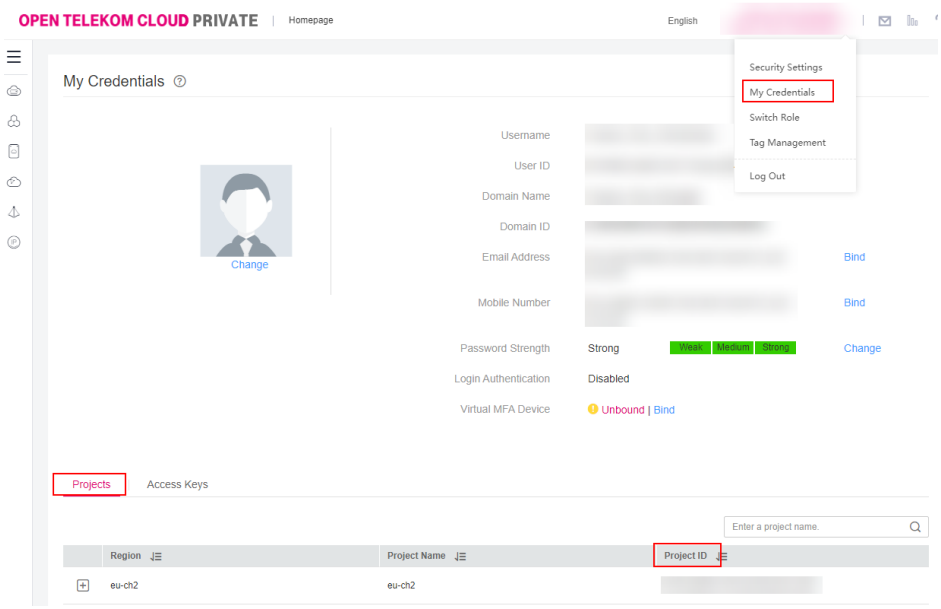
Obtain a Project ID from the Console

To obtain a project ID from the console, perform the following operations:

1. Log in to the management console.
2. Click the username and select **My Credentials** from the drop-down list.

   On the **My Credentials** page, view the project ID (value in the **Project ID** column).

Figure 5-1 Viewing the project ID



6 Change History

Released On	Description
2021-11-12	This issue is the first official release.