# Tag

## Product Documentation

# Contents


Tag ..... 3
- Product Introduction ..... 3
  - Product Overview ..... 3
  - Product Features ..... 5
  - Use Limits ..... 6
  - Tagging-Enabled Services ..... 7
- Operation Guide ..... 9
  - Querying Resources by Tags ..... 9
  - Managing Tags ..... 11
- API Documentation ..... 12
  - Platform product center.(tag) ..... 12
    - version(2018-08-13) ..... 12
      - API overview ..... 12
      - CallingMethod ..... 13
        - signaturev1 ..... 13
        - signaturev3 ..... 21
        - request ..... 30
        - result ..... 31
        - PublicParameter ..... 34
      - Read APIs ..... 36
        - DescribeResourceTags ..... 36
        - DescribeResourceTagsByResourceIds ..... 38
        - DescribeTags ..... 40
      - Write APIs ..... 41
        - AddResourceTag ..... 41
        - CreateTag ..... 43
        - DeleteResourceTag ..... 45
        - DeleteTag ..... 47
        - ModifyResourceTags ..... 49
      - Data structure ..... 50
      - ErrorCode ..... 53

# Product Introduction
# Product Overview

## Introduction

With technological development, the quantity and variety of resources are increasing. Classifying useful resources helps improve work efficiency, so more people prefer using tags to organize their resource files. People can classify tags by keywords, content similarity, and other factors to make future searches and positioning easier. This makes tag management increasingly important.
As the number of cloud platform resources grows, it becomes increasingly difficult for users to manage them. To help users query and manage various resources more efficiently, the cloud platform introduces the tagging service. Tags enable users to classify cloud resources and plan their usage with preset tags. They are words or phrases that serve as metadata to identify and organize cloud resources. Tag limits vary by resource type, but most resources support up to 50 tags.

> Note:
>
> One tag can correspond to multiple resources, and one resource can correspond to multiple tags.

## Glossary

### Tag Key

A tag key is a marker provided by the cloud for identifying resources on the cloud. It is a required field when you set a tag.

### Tag Value

A tag value is a marker provided by the cloud for identifying resources on the cloud. It appears in pairs with a tag key, belongs to the tag key, and can be set to null.

## Workflow

1. First, register and log in to your account and enter the tag management platform.
2. In the tag management platform, you can perform the following two operations:

- Associate existing resources with tags, and create a list of existing resources.
- Create a list of tag keys.

# Product Features

The main features of the product are querying resources and creating tags, as well as tag authorization.

## Querying Resources

For your existing resources in the cloud, you can view and tag them based on their region, type, and instance. The resources viewed will be displayed in a list format. You can name tag keys according to your needs, achieving classification and management of these resources.

## Creating Tags

Creating a tag is equivalent to creating a database locally. It can store all your cloud resource tag information, aiding in future resource management. Creating a tag includes creating both a tag key and tag value.

## Tag Authorization

Tag authorization requires navigating to CAM > Policy Management > Create Custom Policies > Authorize by Tag to complete the authorization process.

# Use Limits

## Tag Key Limits

- Tag keys starting with qcs:, project, etc., are reserved for the system, and creating system-reserved tag keys is prohibited.
- Supports characters in UTF-8 format, including characters, spaces, digits, and special characters + - = ._: / @ .
- The length of a tag key should be between 1 and 127 characters (encoded in UTF-8).
- Tag keys are case sensitive.

## Tag Value Limits

- Supports characters in UTF-8 format, including characters, spaces, digits, and special characters + - = ._: / @ .
- The length of a tag value should be between 1 and 255 characters (encoded in UTF-8).
- Tag values are case sensitive.

## Tag Quantity Limits

- Resource Dimension: A resource can have up to 50 different keys.
- Tag Dimension:
    - A single user can have up to 1,000 different keys.
    - A key can have up to 1,000 values.

# Tagging-Enabled Services

## Compute

| Product Name | Resources |
| --- | --- |
| Cloud Virtual Machine (CVM) | SSH Keys |

## Container

| Product Name | Resources |
| --- | --- |
| Kubernetes Engine (KE) | Container Clusters |

## Storage

| Product Name | Resources |
| --- | --- |
| Cloud Object Storage (COS) | Bucket |
| Cloud File Storage (CFS) | File System |
| Cloud Block Storage (CBS) | CBS Instances |

## Network

| Product Name | Resources |
| --- | --- |
| CLB | Bucket |
| VPC [1] | File System |

> [1] Subnets, route tables, ENI, and security groups in VPC now support the use of Tag.

## Application Security

| Product Name | Resources |
| --- | --- |
| Web Application Firewall | Web Application Firewall |

# Domain Name Service

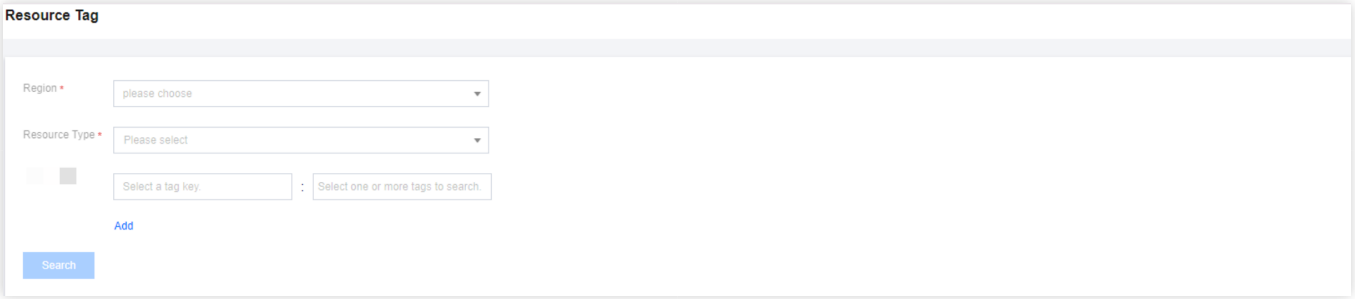
| Product Name | Resources |
| --- | --- |
| VPC DNS | VPC DNS Private Zone |

# Operation Guide
# Querying Resources by Tags

This document describes how to query resources by tag key/value and export the list and detailed operations for editing tag values.

## Querying Tags

1. Log in to the Tag console.
2. In the left sidebar, click Resource Tag to enter the Resource Tag page.
3. On the Resource Tag page, select the following information to set filtering rules.




- Region: The region of the resource to be queried.
- Resource Type: The resource type of the resource to be queried. Only products with tags are supported. For details, see TAG（TAG）Product Introduction> Tags-enabled Products.
- Tag Key/Value: The tag key/value of the resource to be queried. The tag value can be empty, and multiple selections are supported.

4. Click Query, and the corresponding resources are displayed in a list at the bottom of the page to complete the resource query operation.

## Exporting Tags

Check the checkbox in front of the specified tag, select on the right side of the Edit Tag Values, and export the tag list.

# Editing Tag Values

- Batch Editing
  i. Check the checkbox in front of the specified tag, click Edit Tag Value, and open the Edit Tag dialog box.
  ii. In the Edit Tag dialog box, edit the tag value.
  iii. Click OK.
- Editing One by One

  Select the specified tag, click  on the right side of the tag value, and edit the tag value.

# Deleting Tags

- Batch Edit
  i. Check the checkbox in front of the specified tag, click Edit Tag Value, and open the Edit Tag dialog box.
  ii. In the Edit Tag dialog box, click  on the right side of the specified tag value.
  iii. Click OK.
- Editing One by One

  Select the specified tag, click  on the right side of the tag value, and delete the tag value.

# Managing Tags

This document describes the detailed operations for creating and importing tags.

# Create Tag

1. Log in to the Tag console.
2. In the left sidebar, click Tag List to enter the Tag List page.
3. Click Create to open the Add Tag dialog box.
4. Set the tag key and tag value.
5. Click OK.

# Importing Tags

1. Log in to the Tag console.
2. In the left sidebar, click Tag List to enter the Tag List page.
3. Click Import to open the Batch Adding dialog box.
4. Click Download to download the template and add the tag key and tag value locally.
5. Click Select File and select the configured template file.
6. Click Confirm.

# API Documentation
# Platform product center.(tag)
# version(2018-08-13)
# API overview

## API version

V3

## Read APIs

| Interface name | Interface function |
| --- | --- |
| DescribeResourceTags | DescribeResourceTags |
| DescribeResourceTagsByResourceIds | DescribeResourceTagsByResourceIds |
| DescribeTags | DescribeTags |

## Write APIs

| Interface name | Interface function |
| --- | --- |
| AddResourceTag | AddResourceTag |
| CreateTag | CreateTag |
| DeleteResourceTag | DeleteResourceTag |
| DeleteTag | DeleteTag |
| ModifyResourceTags | ModifyResourceTags |

# CallingMethod signaturev1

Converge Cloud authenticates each access request, i.e. each request needs to include authentication information (Signature) in the common parameters to verify the identity of the requester.
The Signature is generated by the security credentials which include SecretId and SecretKey. If you don't have the security credentials yet, go to the API Key page to apply for them; otherwise, you cannot invoke the API.

## 1. Applying for Security Credentials

Before using the API for the first time, go to the API Key page to apply for security credentials.
Security credentials consist of SecretId and SecretKey:

- SecretId is used to identify the API requester.
- SecretKey is used to encrypt the signature string and verify it on the server.
- You must keep your security credentials private and avoid disclosure.

You can apply for the security credentials through the following steps:

1. Log in to the Converge Cloud Console.
2. Go to the Cloud API Key page.
3. On the API Key Management page, click Create Key to create a SecretId/SecretKey pair.

Note: Each account can have up to two pairs of SecretId/SecretKey.

## 2. Generating a Signature

With the SecretId and SecretKey, a signature can be generated. The following describes how to generate a signature:

Assume that the SecretId and SecretKey are:

- SecretId: AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE
- SecretKey: Gu5t9xGARNpq86cd98joQYCN3EXAMPLE

Note: This is just an example. For actual operations, please use your own SecretId and SecretKey.

Take the Cloud Virtual Machine's request to view the instance list (DescribeInstances) as an example. When

you invoke this API, the request parameters may be as follows:

| Parameter name | Description | Parameter value |
|---|---|---|
| Action | Method name | DescribeInstances |
| SecretId | Key ID | AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE |
| Timestamp | Current timestamp | 1465185768 |
| Nonce | Random positive integer | 11886 |
| Region | Region where the instance is located | luzon |
| InstanceIds.0 | ID of the instance to query | ins-09dx96dg |
| Offset | Offset | 0 |
| Limit | Allowed maximum output | 20 |
| Version | API version number | 2017-03-12 |

## 2.1. Sorting Parameters

First, sort all the request parameters in an ascending lexicographical order (ASCII code) by their names. Notes: (1) Parameters are sorted by their names instead of their values; (2) The parameters are sorted based on ASCII code, not in an alphabetical order or by values. For example, InstanceIds.2 should be arranged after InstanceIds.12. You can complete the sorting process using a sorting function in a programming language, such as the ksort function in PHP. The parameters in the example are sorted as follows:

```
{
    'Action' : 'DescribeInstances',
    'InstanceIds.0' : 'ins-09dx96dg',
    'Limit' : 20,
    'Nonce' : 11886,
    'Offset' : 0,
    'Region' : 'luzon',
    'SecretId' : 'AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE',
    'Timestamp' : 1465185768,
    'Version': '2017-03-12',
}
```

When developing in another programming language, you can sort these sample parameters and it will work as long as you obtain the same results.

## 2.2. Concatenating a Request String

This step generates a request string.
Format the request parameters sorted in the previous step into the form of "parameter name"="parameter value". For example, for the Action parameter, its parameter name is "Action" and its parameter value is "DescribeInstances", so it will become Action=DescribeInstances after formatted.
Note: The "parameter value" is the original value but not the value after URL encoding.

Then, concatenate the formatted parameters with "&". The resulting request string is as follows:

```
Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=luzon&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE&Timestamp=1465185768&Version=2017-03-12
```

## 2.3. Concatenating the Signature Original String

This step generates a signature original string.
The signature original string consists of the following parameters:

1. HTTP method: POST and GET modes are supported, and GET is used here for the request. Please note that the method name should be in all capital letters.
2. Request host: the domain name of the request to view the list of instances (DescribeInstances) is cvm.convergecloud.com. The actual request domain name varies by the module to which the API belongs. For more information, see the instructions of the specific API.
3. Request path: The request path in the current version of Cloud API is fixed to / .
4. Request string: the request string generated in the previous step.

The concatenation rule of the signature original string is: Request method + request host + request path + ? + request string

The concatenation result of the example is:

```
GETcvm.convergecloud.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=luzon&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE&Timestamp=1465185768&Version=2017-03-12
```

## 2.4. Generating a Signature String

This step generates a signature string.
First, use the HMAC-SHA1 algorithm to sign thesignature original stringobtained in the previous step, and then encode the generated signature using Base64 to obtain the final signature.

The specific code is as follows with the PHP language being used as an example:

```
$secretKey = 'Gu5t9xGARNpq86cd98joQYCN3EXAMPLE';
$srcStr = 'GETcvm.convergecloud.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit
=20&Nonce=11886&Offset=0&Region=luzon&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPL
E&Timestamp=1465185768&Version=2017-03-12';
$signStr = base64_encode(hash_hmac('sha1', $srcStr, $secretKey, true));
echo $signStr;
```

The final signature is:

```
EliP9YW3pW28FpsEdkXt/+WcGeI=
```

When developing in another programming language, you can sign and verify the original in the example above and it works as long as you get the same results.

# 3. Encoding a Signature String

The generated signature string cannot be directly used as a request parameter and must be URL encoded.

For example, if the signature string generated in the previous step is EliP9YW3pW28FpsEdkXt/+WcGeI= , the final signature string request parameter (Signature) is EliP9YW3pW28FpsEdkXt%2f%2bWcGeI%3d, which will be used to generate the final request URL.

**Note: If your request method is GET, or the request method is POST and the Content-Type is application/x-www-form-urlencoded, then all the request parameter values need to be URL encoded (except the parameter key and the symbol of =) when sending the request. Non-ASCII characters need to be encoded with UTF-8 before URL encoding. **

Note: The network libraries of some programming languages automatically URL encode all parameters, in which case there is no need to URL encode the signature string; otherwise, two rounds of URL encoding will cause the signature to fail.

Note: Other parameter values also need to be encoded using RFC 3986. Use %XY in percent-encoding for special characters such as Chinese characters, where "X" and "Y" are hexadecimal characters (0-9 and uppercase A-F), and using lowercase will cause an error.

# 4. Signature Failure

The following situational error codes for signature failure may occur. Please resolve the errors accordingly.

| Error code | Error description |
| --- | --- |
| AuthFailure.SignatureExpire | The signature is expired |

| Error code | Error description |
| --- | --- |
| AuthFailure.SecretIdNotFound | The key does not exist |
| AuthFailure.SignatureFailure | Signature error |
| AuthFailure.TokenFailure | Token error |
| AuthFailure.InvalidSecretId | Invalid key (not a Cloud API key type) |

# 5. Signature Demo

When calling API 3.0, you are recommended to use the corresponding Converge Cloud SDK 3.0, Swhich encapsulates the signature process, enabling you to focus on only the specific APIs provided by the product when developing. See SDK Center for more information. Currently, the following programming languages are supported:

- Python
- Java
- PHP
- Go
- Node

To further explain the signing process, we will use a programming language to implement the process described above. The request domain name, API and parameter values in the sample are used here. This goal of this example is only to provide additional clarification for the signature process, please see the SDK for actual usage.

The final output URL might be:  https://cvm.convergecloud.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=luzon&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE&Signature=EliP9YW3pW28FpsEdkXt%2F%2BWcGeI%3D&Timestamp=1465185768&Version=2017-03-12

Note: The key in the example is fictitious, and the timestamp is not the current time of the system, so if this URL is opened in the browser or called using commands such as curl, an authentication error will be returned: Signature expired. In order to get a URL that can work properly, you need to replace the SecretId and SecretKey in the example with your real credentials and use the current time of the system as the Timestamp.

Note: In the example below, even if you use the same programming language, the order of the parameters in the URL may be different for each execution. However, the order does not matter, as long as all the parameters are included in the URL and the signature is calculated correctly.

Note: The following code is only applicable to API 3.0. It cannot be directly used in other signature processes. Even with an older API, signature calculation errors may occur due to the differences in details. Please refer to the corresponding documentation.

## Java

```
import java.io.UnsupportedEncodingException;
import java.net.URLEncoder;
import java.util.Random;
import java.util.TreeMap;
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import javax.xml.bind.DatatypeConverter;

public class CloudAPIDemo {
    private final static String CHARSET = "UTF-8";

    public static String sign(String s, String key, String method) throws Exception {
        Mac mac = Mac.getInstance(method);
        SecretKeySpec secretKeySpec = new SecretKeySpec(key.getBytes(CHARSET), mac.getAlgorithm());
        mac.init(secretKeySpec);
        byte[] hash = mac.doFinal(s.getBytes(CHARSET));
        return DatatypeConverter.printBase64Binary(hash);
    }

    public static String getStringToSign(TreeMap<String, Object> params) {
        StringBuilder s2s = new StringBuilder("GETcvm.convergecloud.com/?");
        // When signing, the parameters need to be sorted in lexicographical order. TreeMap is used here
to guarantee the correct order.
        for (String k : params.keySet()) {
            s2s.append(k).append("=").append(params.get(k).toString()).append("&");
        }
        return s2s.toString().substring(0, s2s.length() - 1);
    }

    public static String getUrl(TreeMap<String, Object> params) throws UnsupportedEncodingException
{
        StringBuilder url = new StringBuilder("https://cvm.convergecloud.com/?");
        // There is no requirement for the order of the parameters in the actual request URL.
        for (String k : params.keySet()) {
            // The request string needs to be URL encoded. As the Key is all in English letters, only the valu
e is URL encoded here.
            url.append(k).append("=").append(URLEncoder.encode(params.get(k).toString(), CHARSET)).app
end("&");
        }
        return url.toString().substring(0, url.length() - 1);
```

```
    }

    public static void main(String[] args) throws Exception {
        TreeMap<String, Object> params = new TreeMap<String, Object>(); // TreeMap enables automati
c sorting
        // A random number should be used when actually calling, for example: params.put("Nonce", new
Random().nextInt(java.lang.Integer.MAX_VALUE));
        params.put("Nonce", 11886); // Common parameter
        // The current time of the system should be used when actually calling, for example:   params.put
("Timestamp", System.currentTimeMillis() / 1000);
        params.put("Timestamp", 1465185768); // Common parameter
        params.put("SecretId", "AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE"); // Common parameter
        params.put("Action", "DescribeInstances"); // Common parameter
        params.put("Version", "2017-03-12"); // Common parameter
        params.put("Region", "luzon"); // Common parameter
        params.put("Limit", 20); // Business parameter
        params.put("Offset", 0); // Business parameter
        params.put("InstanceIds.0", "ins-09dx96dg"); // Business parameter
        params.put("Signature", sign(getStringToSign(params), "Gu5t9xGARNpq86cd98joQYCN3EXAMPLE
", "HmacSHA1")); // Common parameter
        System.out.println(getUrl(params));
    }
}
```

## Python

Note: If running in a Python 2 environment, the following requests dependency package must be installed first:  pip install requests .

```
# -*- coding: utf8 -*-
import base64
import hashlib
import hmac
import time

import requests

secret_id = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE"
secret_key = "Gu5t9xGARNpq86cd98joQYCN3EXAMPLE"

def get_string_to_sign(method, endpoint, params):
    s = method + endpoint + "/?"
    query_str = "&".join("%s=%s" % (k, params[k]) for k in sorted(params))
    return s + query_str

def sign_str(key, s, method):
```

```
    hmac_str = hmac.new(key.encode("utf8"), s.encode("utf8"), method).digest()
    return base64.b64encode(hmac_str)

if __name__ == '__main__':
    endpoint = "cvm.convergecloud.com"
    data = {
        'Action' : 'DescribeInstances',
        'InstanceIds.0' : 'ins-09dx96dg',
        'Limit' : 20,
        'Nonce' : 11886,
        'Offset' : 0,
        'Region' : 'luzon',
        'SecretId' : secret_id,
        'Timestamp' : 1465185768, # int(time.time())
        'Version': '2017-03-12'
    }
    s = get_string_to_sign("GET", endpoint, data)
    data["Signature"] = sign_str(secret_key, s, hashlib.sha1)
    print(data["Signature"])
    # n actual invocation would occur here, which may incur fees after success
    # resp = requests.get("https://" + endpoint, params=data)
    # print(resp.url)
```

# signaturev3

Converge Cloud The API will authenticate each access request, meaning that each request must include signature information (Signature) in the public request parameters to verify the requester's identity.
The signature information is generated by the security credentials, which include SecretId and SecretKey . If the user does not have security credentials, please apply for them on the Cloud API Key page, otherwise they cannot call the Cloud API interfaces.

## 1. Apply for security credentials.

Before using the Cloud API for the first time, please go to the Cloud API Key page to apply for a security credential.
Security credentials include SecretId and SecretKey:

- SecretId The security credentials are used to identify the identity of the API caller.
- SecretKey The secret key is used to encrypt and sign the string for server-side verification of the signature.
- Users must strictly protect their security credentials and avoid disclosure them.

The specific steps to apply for security credentials are as follows:

1. Log in to Converge Cloud Administration Center Console
2. Go to the console page for cloud API keys
3. On the Cloud API Key page, click New to create a pair of SecretId/SecretKey

Note: Developer accounts can have up to two pairs SecretId / SecretKey.

## 2. TC3-HMAC-SHA256 signature method

Note: For GET method, only Content-Type: application/x-www-form-urlencoded protocol format is supported. For POST method, Content-Type: application/json and Content-Type: multipart/form-data are currently supported. By default, all business interfaces are supported in json format. Only specific business interfaces are supported in multipart format. In this case, the interface cannot be called in json format. Refer to the specific business interface document description.

The following takes Cloud Virtual Machine querying Region-1 instance list as an example to introduce the signature calculation process step by step. We only use two parameters for the query instance list: Limit and Offset, called using the GET method.

Assume that the SecretId and SecretKey of the user are AKIDz 8krbsJ 5yKBZQpn 74WFkmLPx 3EXAMPLE and Gu 5t9xGARNpq 86cd 98joQYCN3EXAMPLE, respectively.

## 2.1. splice specification request string

Concatenate CanonicalRequest strings in the following format:

```
CanonicalRequest =
  HTTPRequestMethod + '\n' +
  CanonicalURI + '\n' +
  CanonicalQueryString + '\n' +
  CanonicalHeaders + '\n' +
  SignedHeaders + '\n' +
  HashedRequestPayload
```

- HTTPRequestMethod: HTTP request methods (GET, POST), GET in this example;
- CanonicalURI: URI parameter, API 3.0 fixed to forward slash (/);
- CanonicalQueryString: Query string in the URL that initiates the HTTP request. For POST requests, it is fixed as an empty string. For GET requests, it is the question mark (?) in the URL. The following string content, this example value is: Limit=10&Offset=0. Note: CanonicalQueryString needs to be URL encoded.
- CanonicalHeaders: The header information of participating signature, at least including host and content-type, can also be added with customized header to improve the uniqueness and security of its own request. Splicing rules: 1) head key and value are uniformly converted to lowercase, and the first and last spaces are removed, and spliced according to the format of key:value\n;2) multiple heads are spliced according to the dictionary sorting of head key (lowercase). In this example: content-type: application/x-www-form-urlencoded\nhost: cvm.convergecloud.com\n
- SignedHeaders: The header information involved in the signature indicates which headers participate in the signature of this request, and the header content contained in CanonicalHeaders is one-to-one corresponding. content-type and host are required headers. constructed as: 1) The head key is uniformly converted to lowercase; 2) Multiple header keys (lowercase) are concatenated according to dictionary sorting and separated by semicolons (;). In this case: content-type;host
- HashedRequestPayload: The hash value of the request body, calculated as Lowercase(HexEncode(Hash.SHA256(RequestPayload))), SHA256 hash the entire payload of the HTTP request body, then hexadecimal encoding, and finally converting the encoding string to lowercase letters. Note: For GET requests, RequestPayload is fixed as an empty string. For POST requests, RequestPayload is the payload of the HTTP request body.

According to the above rules, the resulting canonical request string in the example is as follows (for clarity, \nnewlines are replaced by a new line printed separately):

```
GET
```

```
/
Limit=10&Offset=0
content-type:application/x-www-form-urlencoded
host:cvm.convergecloud.com

content-type;host
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

## 2.2. Concatenate the string to be signed

Concatenate the string to be signed in the following format:

```
StringToSign =
  Algorithm + \n +
  RequestTimestamp + \n +
  CredentialScope + \n +
  HashedCanonicalRequest
```

- Algorithm: Signature algorithm, currently fixed as TC3-HMAC-SHA256;
- RequestTimestamp: Request timestamp, i.e. the X-TC-Timestamp value of the request header. The request in the example above is 1539084154;
- CredentialScope: Credential range in the format Date/service/tc3_request, containing date, requested service and termination string (tc3_request). **Date is the date of UTC standard time, the value needs to be consistent with UTC standard time converted by common parameter X-TC-Timestamp **;service is the product name, which must be consistent with the product domain name called, such as cvm. The above example request, the value is 2018-10-09/cvm/tc3_request;
- HashedCanonicalRequest: The hash value of the canonical request string obtained by concatenating the above steps is calculated by Lowercase(HexEncode(Hash.SHA256(CanonicalRequest))).

note:

> 1. Date must be calculated from timestamp X-TC-Timestamp and time zone UTC+0. Adding local time zone information to the system, such as East Zone 8, will cause calls to succeed during the day and night, but calls to fail in the early hours of the morning. Assuming the timestamp is 1551113065, the time in the East Eight District is 2019-02-26 00:44:25, but the calculated Date with UTC+0 should be 2019-02-25 instead of 2019-02-26.
> 2. Timestamp must be the current system time and ensure that the system time and standard time are synchronized. If the difference exceeds five minutes, it will fail. If it is not synchronized with the standard time for a long time, it may cause the request to fail after running for a period of time (return signature expiration error).

According to the above rules, the string to be signed in the example is as follows (for clarity,\nnewlines are

replaced by new lines printed):

```
TC3-HMAC-SHA256
1539084154
2018-10-09/cvm/tc3_request
91c9c192c14460df6c1ffc69e34e6c5e90708de2a6d282cccf957dbf1aa7f3a7
```

## 2.3. calculate the signature

1. Compute the derived signature key, pseudocode as follows

```
SecretKey = "Gu5t9xGARNpq86cd98joQYCN3EXAMPLE"
SecretDate = HMAC_SHA256("TC3" + SecretKey, Date)
SecretService = HMAC_SHA256(SecretDate, Service)
SecretSigning = HMAC_SHA256(SecretService, "tc3_request")
```

- SecretKey: Original SecretKey;
- Date: That is, the Date field information in Credential, as shown in the example above, is 2018-10-09.;
- Service: That is, the Service field information in Credential, as shown in the example above, is cvm;

2. Calculate the signature, pseudocode as follows

```
Signature = HexEncode(HMAC_SHA256(SecretSigning, StringToSign))
```

- SecretSigning: That is, the derivative signature key calculated above;
- StringToSign: That is, the character string to be signed calculated in step 2;

## 2.4. Splicing Authorization

Splice as follows Authorization:

```
Authorization =
    Algorithm + ' ' +
    'Credential=' + SecretId + '/' + CredentialScope + ', ' +
    'SignedHeaders=' + SignedHeaders + ', '
    'Signature=' + Signature
```

- Algorithm: Signature method, fixed as TC3-HMAC-SHA256;
- SecretId: SecretId in key pair;
- CredentialScope: See above, Scope of Voucher;
- SignedHeaders: See above, header information for participating signatures;
- Signature: signature value

According to the above rules, the values obtained in the example are:

```
TC3-HMAC-SHA256 Credential=AKIDEXAMPLE/Date/service/tc3_request, SignedHeaders=content-type;host, Signature=5da7a33f6993f0614b047e5df4582db9e9bf4672ba50567dba16c6ccf174c474
```

The final complete invocation information is as follows:

```
https://cvm.convergecloud.com/?Limit=10&Offset=0

Authorization: TC3-HMAC-SHA256 Credential=AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE/2018-10-09/cvm/tc3_request, SignedHeaders=content-type;host, Signature=5da7a33f6993f0614b047e5df4582db9e9bf4672ba50567dba16c6ccf174c474
Content-Type: application/x-www-form-urlencoded
Host: cvm.convergecloud.com
X-TC-Action: DescribeInstances
X-TC-Version: 2017-03-12
X-TC-Timestamp: 1539084154
X-TC-Region: luzon
```

# 3. signing failed

According to the actual situation, there are the following signature failure error codes, please handle according to the actual situation

| error code | error description |
| --- | --- |
| AuthFailure.SignatureExpire | signature expiration |
| AuthFailure.SecretIdNotFound | key is not present |
| AuthFailure.SignatureFailure | signature error |
| AuthFailure.TokenFailure | token error |
| AuthFailure.InvalidSecretId | Illegal key (not cloud API key type) |

# 4. Signature Demo

Java

```
import java.io.BufferedReader;
import java.io.InputStream;
```

```
import java.io.InputStreamReader;
import java.net.URL;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Map;
import java.util.TimeZone;
import java.util.TreeMap;
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import javax.net.ssl.HttpsURLConnection;
import javax.xml.bind.DatatypeConverter;

import org.apache.commons.codec.digest.DigestUtils;

public class CloudAPITC3Demo {
    private final static String CHARSET = "UTF-8";
    private final static String ENDPOINT = "cvm.convergecloud.com";
    private final static String PATH = "/";
    private final static String SECRET_ID = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE";
    private final static String SECRET_KEY = "Gu5t9xGARNpq86cd98joQYCN3EXAMPLE";
    private final static String CT_X_WWW_FORM_URLENCODED = "application/x-www-form-urlencode
d";
    private final static String CT_JSON = "application/json";
    private final static String CT_FORM_DATA = "multipart/form-data";

    public static byte[] sign256(byte[] key, String msg) throws Exception {
        Mac mac = Mac.getInstance("HmacSHA256");
        SecretKeySpec secretKeySpec = new SecretKeySpec(key, mac.getAlgorithm());
        mac.init(secretKeySpec);
        return mac.doFinal(msg.getBytes(CHARSET));
    }

    public static void main(String[] args) throws Exception {
        String service = "cvm";
        String host = "cvm.convergecloud.com";
        String region = "luzon";
        String action = "DescribeInstances";
        String version = "2017-03-12";
        String algorithm = "TC3-HMAC-SHA256";
        String timestamp = "1539084154";
        //String timestamp = String.valueOf(System.currentTimeMillis() / 1000);
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
        // Pay attention to the time zone, otherwise it is easy to make mistakes.
        sdf.setTimeZone(TimeZone.getTimeZone("UTC"));
        String date = sdf.format(new Date(Long.valueOf(timestamp + "000")));

        // ************* Step 1: Assemble the specification request string *************
```

```
        String httpRequestMethod = "GET";
        String canonicalUri = "/";
        String canonicalQueryString = "Limit=10&Offset=0";
        String canonicalHeaders = "content-type:application/x-www-form-urlencoded\n" + "host:" + host
+ "\n";
        String signedHeaders = "content-type;host";
        String hashedRequestPayload = DigestUtils.sha256Hex("");
        String canonicalRequest = httpRequestMethod + "\n" + canonicalUri + "\n" + canonicalQueryStri
ng + "\n"
                + canonicalHeaders + "\n" + signedHeaders + "\n" + hashedRequestPayload;
        System.out.println(canonicalRequest);

        // ************* Step 2: Stitching the string to be signed *************
        String credentialScope = date + "/" + service + "/" + "tc3_request";
        String hashedCanonicalRequest = DigestUtils.sha256Hex(canonicalRequest.getBytes(CHARSET));
        String stringToSign = algorithm + "\n" + timestamp + "\n" + credentialScope + "\n" + hashedCan
onicalRequest;
        System.out.println(stringToSign);

        // ************* Step 3: Calculate the signature *************
        byte[] secretDate = sign256(("TC3" + SECRET_KEY).getBytes(CHARSET), date);
        byte[] secretService = sign256(secretDate, service);
        byte[] secretSigning = sign256(secretService, "tc3_request");
        String signature = DatatypeConverter.printHexBinary(sign256(secretSigning, stringToSign)).toLow
erCase();
        System.out.println(signature);

        // ************* Step 4: Splicing Authorization *************
        String authorization = algorithm + " " + "Credential=" + SECRET_ID + "/" + credentialScope + ", "
                + "SignedHeaders=" + signedHeaders + ", " + "Signature=" + signature;
        System.out.println(authorization);

        TreeMap<String, String> headers = new TreeMap<String, String>();
        headers.put("Authorization", authorization);
        headers.put("Host", host);
        headers.put("Content-Type", CT_X_WWW_FORM_URLENCODED);
        headers.put("X-TC-Action", action);
        headers.put("X-TC-Timestamp", timestamp);
        headers.put("X-TC-Version", version);
        headers.put("X-TC-Region", region);
    }
}
```

## Python

```
# -*- coding: utf-8 -*-
```

```
import hashlib, hmac, json, os, sys, time
from datetime import datetime

# key parameters
secret_id = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE"
secret_key = "Gu5t9xGARNpq86cd98joQYCN3EXAMPLE"

service = "cvm"
host = "cvm.convergecloud.com"
endpoint = "https://" + host
region = "luzon"
action = "DescribeInstances"
version = "2017-03-12"
algorithm = "TC3-HMAC-SHA256"
timestamp = 1539084154
date = datetime.utcfromtimestamp(timestamp).strftime("%Y-%m-%d")
params = {"Limit": 10, "Offset": 0}

# ************* Step 1: Assemble the specification request string *************
http_request_method = "GET"
canonical_uri = "/"
canonical_querystring = "Limit=10&Offset=0"
ct = "x-www-form-urlencoded"
payload = ""
if http_request_method == "POST":
    canonical_querystring = ""
    ct = "json"
    payload = json.dumps(params)
canonical_headers = "content-type:application/%s\nhost:%s\n" % (ct, host)
signed_headers = "content-type;host"
hashed_request_payload = hashlib.sha256(payload.encode("utf-8")).hexdigest()
canonical_request = (http_request_method + "\n" +
                     canonical_uri + "\n" +
                     canonical_querystring + "\n" +
                     canonical_headers + "\n" +
                     signed_headers + "\n" +
                     hashed_request_payload)
print(canonical_request)

# ************* Step 2: Stitching the string to be signed *************
credential_scope = date + "/" + service + "/" + "tc3_request"
hashed_canonical_request = hashlib.sha256(canonical_request.encode("utf-8")).hexdigest()
string_to_sign = (algorithm + "\n" +
                  str(timestamp) + "\n" +
                  credential_scope + "\n" +
                  hashed_canonical_request)
print(string_to_sign)
```

```
# ************* Step 3: Calculate the signature *************
# Compute signature digest function
def sign(key, msg):
    return hmac.new(key, msg.encode("utf-8"), hashlib.sha256).digest()
secret_date = sign(("TC3" + secret_key).encode("utf-8"), date)
secret_service = sign(secret_date, service)
secret_signing = sign(secret_service, "tc3_request")
signature = hmac.new(secret_signing, string_to_sign.encode("utf-8"), hashlib.sha256).hexdigest()
print(signature)

# ************* Step 4: Splicing Authorization *************
authorization = (algorithm + " " +
                 "Credential=" + secret_id + "/" + credential_scope + ", " +
                 "SignedHeaders=" + signed_headers + ", " +
                 "Signature=" + signature)
print(authorization)

# Common parameters added to request header
headers = {
    "Authorization": authorization,
    "Host": host,
    "Content-Type": "application/%s" % ct,
    "X-TC-Action": action,
    "X-TC-Timestamp": str(timestamp),
    "X-TC-Version": version,
    "X-TC-Region": region,
}
```

# request

## 1. Service Address

Region refers to the geographical area of a physical data center. Converge Cloud ensures complete isolation between different regions, providing maximum stability and fault tolerance. To reduce access latency and improve download speed, it is recommended to choose the region closest to your customers.

You can view the complete region list by using the API interface "DescribeRegions".

## 2. Communications Protocol

All the Converge Cloud APIs communicate via HTTPS, providing highly secure communication tunnels.

## 3. Request Methods

Supported HTTP request methods:

- POST (recommended)
- GET

PThe Content-Type types supported by POST requests:

- application/json (recommended). The TC3-HMAC-SHA256 signature algorithm must be used.
- application/x-www-form-urlencoded. The HmacSHA1 or HmacSHA256 signature algorithm must be used.
- multipart/form-data (only supported by certain APIs). You must use TC3-HMAC-SHA256 to calculate the signature.

The size of a GET request packet is up to 32 KB. The size of a POST request is up to 1 MB when the HmacSHA1 or HmacSHA256 signature algorithm is used, and up to 10 MB when TC3-HMAC-SHA256 is used.

## 4. Character Encoding

Only UTF-8 encoding is used.

# result

## Response for Successful Requests

For example, when calling CAM API (version: 2017-03-12) to view the status of instances (DescribeInstancesStatus), if the request has succeeded, you may see the response as shown below:


```
{
    "Response": {
        "TotalCount": 0,
        "InstanceStatusSet": [],
        "RequestId": "b5b41468-520d-4192-b42f-595cc34b6c1c"
    }
}
```


- The API will return Response, which contains RequestId, as long as it processes the request. It does not matter if the request is successful or not.
- RequestId is the unique ID of an API request. Contact us with this ID when an exception occurs.
- Except for the fixed fields, all fields are action-specified. For the definitions of action-specified fields, see the corresponding API documentation. In this example, TotalCount and InstanceStatusSet are the fields specified by the API DescribeInstancesStatus. 0 TotalCount means that the requester owns 0 CVM instance so the InstanceStatusSet is empty.

## Response for Failed Requests

If the request has failed, you may see the response as shown below:


```
{
    "Response": {
        "Error": {
            "Code": "AuthFailure.SignatureFailure",
            "Message": "The provided credentials could not be validated. Please check your signature is cor
rect."
        },
        "RequestId": "ed93f3cb-f35e-473f-b9f3-0d451b8b79c6"
    }
}
```


- The presence of the Error field indicates that the request has failed. A response for a failed request will include Error, Code and Message fields.

- Code is the code of the error that helps you identify the cause and solution. There are two types of error codes so you may find the code in either common error codes or API-specified error codes.
- Message explains the cause of the error. Note that the returned messages are subject to service updates. The information the messages provide may not be up-to-date and should not be the only source of reference.
- RequestId is the unique ID of an API request. Contact us with this ID when an exception occurs.

# Common Error Codes

If there is an Error field in the response, it means that the API call failed. The Code field in Error indicates the error code. The following table lists the common error codes that all actions can return.

| Error Code | Description |
| --- | --- |
| AuthFailure.InvalidSecretId | Invalid key (not a API key type). |
| AuthFailure.MFAFailure | MFA failed. |
| AuthFailure.SecretIdNotFound | The key does not exist. |
| AuthFailure.SignatureExpire | Signature expired. |
| AuthFailure.SignatureFailure | Signature error. |
| AuthFailure.TokenFailure | Token error. |
| AuthFailure.UnauthorizedOperation | The request does not have CAM authorization. |
| DryRunOperation | DryRun Operation. It means that the request would have succeeded, but the DryRun parameter was used. |
| FailedOperation | Operation failed. |
| InternalError | Internal error. |
| InvalidAction | The API does not exist. |
| InvalidParameter | Incorrect parameter. |
| InvalidParameterValue | Invalid parameter value. |
| LimitExceeded | Quota limit exceeded. |
| MissingParameter | A parameter is missing. |
| NoSuchVersion | The API version does not exist. |

| Error Code | Description |
| --- | --- |
| RequestLimitExceeded | The number of requests exceeds the frequency limit. |
| ResourceInUse | Resource is in use. |
| ResourceInsufficient | Insufficient resource. |
| ResourceNotFound | The resource does not exist. |
| ResourceUnavailable | Resource is unavailable. |
| UnauthorizedOperation | Unauthorized operation. |
| UnknownParameter | Unknown parameter. |
| UnsupportedOperation | Unsupported operation. |
| UnsupportedProtocol | hHTTPS request method error. Only GET and POST requests are supported. |
| UnsupportedRegion | API does not support the requested region. |

# PublicParameter

Common parameters are parameters used to identify users and interface authentication purposes. If not necessary, these parameters are not described in the separate interface document of each interface, but these parameters need to be carried in each request before the request can be normally initiated.

## signature method v3

When using TC3-HMAC-SHA256 signature method, the common parameters need to be uniformly placed in the HTTP Header request header, as follows:

| parameter name | type | required | described |
|---|---|---|---|
| X-TC-Action | String | Yes | The name of the operation interface. Refer to the description of the public parameter Action in the input parameters of the interface documentation for the value. For example, the value for the query instance list interface of the cloud server is DescribeInstances. |
| X-TC-Region | String | Yes | region parameters, Data used to identify which region you wish to operate on. The region value accepted by the interface refers to the description of the input parameter common parameter Region in the interface document. Note: Some interfaces do not need to pass this parameter, which is specifically stated in the interface documentation. In this case, even passing this parameter will not take effect. |
| X-TC-Timestamp | Integer | Yes | The current UNIX timestamp, which records the time when the API request was initiated. for example 1529223702. Note: If the time difference from the server exceeds 5 minutes, it will cause a signature expiration error. |
| X-TC-Version | String | Yes | Version of API for operation. Refer to the description of the public parameter Version in the interface document for the value. For example, the version of the cloud server. 2017-03-12. |
| Authorization | String | Yes | HTTP The standard authentication header field is "Authorization". For example: TC3-HMAC-SHA256 Credential=AKIDEXAMPLE/Date/service/tc3_request, SignedHeaders=content-type;host, Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024 Here,<br>- TC3-HMAC-SHA256: Signature method, currently fixed as this value,<br>- Credential: Credential: Signature credential; AKIDEXAMPLE is the SecretId; Date is a date in UTC time, and this value must match the value of X-TC-Timestamp (a common parameter) in UTC time format; service is the name of the product/service, and is generally a domain name prefix. For example,cvm;<br>- SignedHeaders: The headers that contains the authentication information; content-type and host are the required headers;<br>- Signature: Signature digest. |
| X-TC-Token | String | No | The token used for temporary certificates needs to be used in conjunction with a temporary key. Temporary keys and tokens need to be obtained by calling the access management service interface. Long-term keys do not require tokens. |

## signature method v1

When using HmacSHA1 and HmacSHA256 signature methods, common parameters need to be placed uniformly in the request string, as follows

| parameter name | type | required | described |
| --- | --- | --- | --- |
| Action | String | Yes | The name of the interface to be operated. The value refers to the explanation of the public parameter Action in the interface documentation. For example, for the cloud server's query instance list interface, the value is DescribeInstances . |
| Region | String | Yes | The region parameter is used to identify which region's data you want to operate on. The region value accepted by the interface is specified in the input parameter of the public parameter Region in the interface documentation. Note: Some interfaces do not require this parameter to be passed, which will be specifically mentioned in the interface documentation. Even if this parameter is passed, it will not take effect in this case. |
| Timestamp | Integer | Yes | The current UNIX timestamp, which can record the time when the API request is initiated. For example, 1529223702. If the difference between the current time is too large, it will cause a signature expiration error. |
| Nonce | Integer | Yes | A random positive integer combined with Timestamp is used to prevent replay attacks. |
| SecretId | String | Yes | The SecretId is an identifier applied for the cloud API key to identify the user's identity. A SecretId corresponds to a unique SecretKey , and the SecretKey is used to generate the request signature Signature . |
| Signature | String | Yes | Request signature is used to verify the legitimacy of this request. Users need to calculate it based on the actual input parameters. The specific calculation method can be found in the interface authentication documentation. |
| Version | String | Yes | The version of the API being operated on. The value refers to the description of the public parameter Version in the interface documentation. For example, the version of the cloud server. 2017-03-12. |
| SignatureMethod | String | No | Signature method, currently supports HmacSHA256 and HmacSHA1. HmacSHA256 algorithm is used to verify signatures only if this parameter is specified as HmacSHA 256, otherwise HmacSHA1 is used to verify signatures. |
| Token | String | No | Token used for temporary certificate needs to be used together with temporary key. The temporary key and Token need to be obtained from the Access Management service invocation interface. Long-term keys do not require tokens. |

# zone list

A region (Region) refers to the geographical area of a physical data center. Converge Cloud Translation: To deliver verification that completely isolates different regions, ensuring maximum stability and fault tolerance between regions. In order to reduce access latency and improve download speed, it is recommended that you choose the region closest to your customers.

You can view the complete region list by using the API interface Query Region List.

# Read APIs
# DescribeResourceTags

## 1. Interface Description

Interface request domain name: tag.api3.convergecloud.com.

This API is used to query the tags associated with a resource.

Default interface request rate limit: 20 times per second.

Interface update time: 2020-05-19 06:20:33.

The interface is both signature verification and authentication.

## 2. Input Parameters

The following list of request parameters only lists interface request parameters and some public parameters, For a complete list of public parameters see public request parameters.

| Parameter name | Required | Allow NULL | Type | Description |
|---|---|---|---|---|
| Action | Yes | No | String | Public parameter, The value of this interface: DescribeResourceTags |
| Version | Yes | No | String | Public parameter, The value of this interface: 2018-08-13 |
| Region | No | No | String | Public parameter,Regional information This interface does not need to pass this parameter. |
| CreateUin | No | No | Uint64 | Creator UIN.<br>Example: 100004611496 |
| ResourceRegion | No | No | String | Region where a resource is located.<br>Example: ap-region-y |
| ServiceType | No | No | String | Suitable Business Type<br>Example: ckafka |
| ResourcePrefix | No | No | String | Resource Prefix<br>Example: ckafkaId |
| ResourceId | No | No | String | Resource UID.<br>Example: ckafka-o85jq584 |
| Offset | No | No | Uint64 | Offset value for data. Default value: 0. The value of Offset must be an integer multiple of the value of Limit.<br>Example: 0 |
| Limit | No | No | Uint64 | Size of each page. Default value: 15.<br>Example: 15 |
| CosResourceId | No | No | Uint64 | Specifies whether it is an ID of a resource in Cloud Object Storage.<br>Example: 0 |

## 3. Output Parameters

| Parameter name | Type | Description |
|---|---|---|
| TotalCount | Uint64 | Number of returned results.<br>Example: 320 |
| Offset | Uint64 | Offset value for data.<br>Example: 0 |
| Limit | Uint64 | Page size.<br>Example: 15 |

| Parameter name | Type | Description |
| --- | --- | --- |
| Rows | Array of TagResource | Resource tag.<br>Example:View |
| RequestId | String | Unique request id, Each request will return. The RequestId of the request needs to be provided when locating the problem. |

# 4. Error Code

There is no error code related to business logic for this interface, For other error codes, see public ErrorCode.

# DescribeResourceTagsByResourceIds

## 1. Interface Description

Interface request domain name: tag.api3.convergecloud.com.

API for querying tag key-value pairs associated with existing resources.

Default interface request rate limit: 20 times per second.

Interface update time: 2024-12-06 17:00:18.

The interface is both signature verification and authentication.

## 2. Input Parameters

The following list of request parameters only lists interface request parameters and some public parameters, For a complete list of public parameters see public request parameters.

| Parameter name | Required | Allow NULL | Type | Description |
|---|---|---|---|---|
| Action | Yes | No | String | Public parameter, The value of this interface: DescribeResourceTagsByResourceIds |
| Version | Yes | No | String | Public parameter, The value of this interface: 2018-08-13 |
| Region | No | No | String | Public parameter,Regional information This interface does not need to pass this parameter. |
| ServiceType | Yes | No | String | Suitable Business Type<br>Example: cvm |
| ResourcePrefix | Yes | No | String | Resource Prefix<br>Example: instance |
| ResourceIds | Yes | No | Array of String | Resource UID.<br>Example: ins-qcmain |
| ResourceRegion | Yes | No | String | Region where a resource is located.<br>Example: ap-region-x |
| Offset | No | No | Uint64 | Offset value for data. Default value: 0. The value of Offset must be an integer multiple of the value of Limit.<br>Example: 0 |
| Limit | No | No | Uint64 | Size of each page. Default value: 15.<br>Example: 15 |

## 3. Output Parameters

| Parameter name | Type | Description |
|---|---|---|
| TotalCount | Uint64 | Number of returned results.<br>Example: 15 |
| Offset | Uint64 | Offset value for data.<br>Example: 0 |
| Limit | Uint64 | Page size.<br>Example: 15 |
| Tags | Array of TagResource | Tag List<br>Example:View |
| RequestId | String | Unique request id, Each request will return. The RequestId of the request needs to be provided when locating the problem. |

# 4. Error Code

The following lists only the error codes related to the interface business logic, For other error codes, see public ErrorCode.

| Error Code | Description |
| --- | --- |
| InvalidParameterValue.RegionInvalid | Region error. |
| InvalidParameterValue.ResourcePrefixInvalid | Resource prefix error. |
| InvalidParameterValue.ServiceTypeInvalid | Service type error. |
| InvalidParameterValue.UinInvalid | Uin parameter is invalid. |
| InvalidParameterValue.ResourceIdSizeInvalid | The number of resource exceeds the upper limit (50). |

# DescribeTags

## 1. Interface Description

Interface request domain name: tag.api3.convergecloud.com.

API for querying an existing tag list.

Default interface request rate limit: 20 times per second.

Interface update time: 2024-12-06 17:01:38.

The interface is both signature verification and authentication.

## 2. Input Parameters

The following list of request parameters only lists interface request parameters and some public parameters, For a complete list of public parameters see public request parameters.

| Parameter name | Required | Allow NULL | Type | Description |
|---|---|---|---|---|
| Action | Yes | No | String | Public parameter, The value of this interface: DescribeTags |
| Version | Yes | No | String | Public parameter, The value of this interface: 2018-08-13 |
| Region | No | No | String | Public parameter,Regional information This interface does not need to pass this parameter. |
| TagKey | No | No | String | Tag key. If a tag key exists, a tag value exists. If not, a tag value does not exist. If you do not specify this parameter, the system queries all tags of the user.<br>Example: TagKey_cvm |
| TagValue | No | No | String | Tag value. If a tag value exists, a tag key exists. If not, a tag key does not exist. If you do not specify this parameter, the system queries all tags of the user.<br>Example: TagValue_cvm |
| Offset | No | No | Uint64 | Offset value for data. Default value: 0. The value of Offset must be an integer multiple of the value of Limit.<br>Example: 0 |
| Limit | No | No | Uint64 | Size of each page. Default value: 15.<br>Example: 15 |
| CreateUin | No | No | Uint64 | UIN of a tag creator. If you do not specify this parameter, all UINs are used as query conditions.<br>Example: 121 |
| TagKeys | No | No | Array of String | Tag key array. If a tag key exists, tag values exist. If not, tag values do not exist. If you do not specify this parameter, the system queries all tags of the user. If the value of TagKeys is the same as the value of TagKey, the system only returns the value of TagKeys.<br>Example: Key |

## 3. Output Parameters

| Parameter name | Type | Description |
|---|---|---|
| TotalCount | Uint64 | Number of returned results.<br>Example: 1 |
| Offset | Uint64 | Offset value for data.<br>Example: 0 |
| Limit | Uint64 | Page size.<br>Example: 15 |
| Tags | Array of TagWithDelete | Tag List<br>Example:View |
| RequestId | String | Unique request id, Each request will return. The RequestId of the request needs to be provided when locating the problem. |

## 4. Error Code

The following lists only the error codes related to the interface business logic, For other error codes, see public ErrorCode.

| Error Code | Description |
|---|---|
| InvalidParameterValue.UinInvalid | Uin parameter is invalid. |

# Write APIs
# AddResourceTag

## 1. Interface Description

Interface request domain name: tag.api3.convergecloud.com.

This API is used to associate resources with tags.

Default interface request rate limit: 20 times per second.

Interface update time: 2024-12-06 16:56:29.

The interface is both signature verification and authentication.

## 2. Input Parameters

The following list of request parameters only lists interface request parameters and some public parameters, For a complete list of public parameters see public request parameters.

| Parameter name | Required | Allow NULL | Type | Description |
|---|---|---|---|---|
| Action | Yes | No | String | Public parameter, The value of this interface: AddResourceTag |
| Version | Yes | No | String | Public parameter, The value of this interface: 2018-08-13 |
| Region | No | No | String | Public parameter,Regional information This interface does not need to pass this parameter. |
| TagKey | Yes | No | String | Tag key<br>Example: TagKey_cvm |
| TagValue | Yes | No | String | Tag value<br>Example: TagValue_cvm |
| Resource | Yes | No | String | Resource description in 6-segment format.<br>Example: qcs::cvm:ap-region-x:uin/100004603715:instance/ins-qcmain |

# 3. Output Parameters

| Parameter name | Type | Description |
| --- | --- | --- |
| RequestId | String | Unique request id, Each request will return. The RequestId of the request needs to be provided when locating the problem. |

# 4. Error Code

The following lists only the error codes related to the interface business logic, For other error codes, see public ErrorCode.

| Error Code | Description |
| --- | --- |
| InvalidParameterValue.ResourceDescriptionError | Invalid resource description. |
| LimitExceeded.TagKey | The number of created tag keys reaches the limit of 1,000. |
| LimitExceeded.TagValue | Each tag key can have up to 1,000 values. |

# CreateTag

## 1. Interface Description

Interface request domain name: tag.api3.convergecloud.com.

This API is used to create a tag key and tag value pair.

Default interface request rate limit: 20 times per second.

Interface update time: 2024-12-06 16:57:56.

The interface is both signature verification and authentication.

## 2. Input Parameters

The following list of request parameters only lists interface request parameters and some public parameters, For a complete list of public parameters see public request parameters.

| Parameter name | Required | Allow NULL | Type | Description |
|---|---|---|---|---|
| Action | Yes | No | String | Public parameter, The value of this interface: CreateTag |
| Version | Yes | No | String | Public parameter, The value of this interface: 2018-08-13 |
| Region | No | No | String | Public parameter,Regional information This interface does not need to pass this parameter. |
| TagKey | Yes | No | String | Tag key<br>Example: TagKey_cvm |
| TagValue | Yes | No | String | Tag value<br>Example: TagValue_cvm |

## 3. Output Parameters

| Parameter name | Type | Description |
|---|---|---|

| Parameter name | Type | Description |
| --- | --- | --- |
| RequestId | String | Unique request id, Each request will return. The RequestId of the request needs to be provided when locating the problem. |

# 4. Error Code

The following lists only the error codes related to the interface business logic, For other error codes, see public ErrorCode.

| Error Code | Description |
| --- | --- |
| LimitExceeded.TagKey | The number of created tag keys reaches the limit of 1,000. |
| LimitExceeded.TagValue | Each tag key can have up to 1,000 values. |
| InvalidParameterValue.ReservedTagKey | Creation of system-reserved tag keys is not allowed. |
| InvalidParameterValue.TagKeyCharacterIllegal | Tag key contains illegal characters. |
| InvalidParameterValue.TagKeyEmpty | Tag key cannot be empty. |
| InvalidParameterValue.TagKeyLengthExceeded | Tag key length exceeds limit. |
| InvalidParameterValue.TagValueCharacterIllegal | Tag value contains illegal characters. |
| InvalidParameterValue.TagValueLengthExceeded | Tag value length exceeds limit. |
| ResourceInUse.TagDuplicate | Tag already exists. |

# DeleteResourceTag

## 1. Interface Description

Interface request domain name: tag.api3.convergecloud.com.

This API is used to unassociate tags and resources.

Default interface request rate limit: 20 times per second.

Interface update time: 2025-04-29 16:16:05.

The interface is both signature verification and authentication.

## 2. Input Parameters

The following list of request parameters only lists interface request parameters and some public parameters, For a complete list of public parameters see public request parameters.

| Parameter name | Required | Allow NULL | Type | Description |
|---|---|---|---|---|
| Action | Yes | No | String | Public parameter, The value of this interface: DeleteResourceTag |
| Version | Yes | No | String | Public parameter, The value of this interface: 2018-08-13 |
| Region | No | No | String | Public parameter,Regional information This interface does not need to pass this parameter. |
| TagKey | Yes | No | String | Tag key<br>Example: key |
| Resource | Yes | No | String | Resource description in 6-segment format.<br>Example: qcs::cvm:ap-region-x:uin/100004603715:instance/qcmain |

## 3. Output Parameters

| Parameter name | Type | Description |
|---|---|---|

| Parameter name | Type | Description |
| --- | --- | --- |
| RequestId | String | Unique request id, Each request will return. The RequestId of the request needs to be provided when locating the problem. |

# 4. Error Code

The following lists only the error codes related to the interface business logic, For other error codes, see public ErrorCode.

| Error Code | Description |
| --- | --- |
| ResourceNotFound.AttachedTagKeyNotFound | The tag key associated with the resource doesn't exist. |
| InvalidParameterValue.ResourceDescriptionError | Invalid resource description. |

# DeleteTag

## 1. Interface Description

Interface request domain name: tag.api3.convergecloud.com.

This API is used to delete a tag key and tag value pair.

Default interface request rate limit: 20 times per second.

Interface update time: 2024-12-06 16:59:23.

The interface is both signature verification and authentication.

## 2. Input Parameters

The following list of request parameters only lists interface request parameters and some public parameters, For a complete list of public parameters see public request parameters.

| Parameter name | Required | Allow NULL | Type | Description |
|---|---|---|---|---|
| Action | Yes | No | String | Public parameter, The value of this interface: DeleteTag |
| Version | Yes | No | String | Public parameter, The value of this interface: 2018-08-13 |
| Region | No | No | String | Public parameter,Regional information This interface does not need to pass this parameter. |
| TagKey | Yes | No | String | Tag key that is required to delete.<br>Example: TagKey_cvm |
| TagValue | Yes | No | String | Tag value that is required to delete.<br>Example: TagValue_cvm |

## 3. Output Parameters

| Parameter name | Type | Description |
|---|---|---|

| Parameter name | Type | Description |
| --- | --- | --- |
| RequestId | String | Unique request id, Each request will return. The RequestId of the request needs to be provided when locating the problem. |

# 4. Error Code

The following lists only the error codes related to the interface business logic, For other error codes, see public ErrorCode.

| Error Code | Description |
| --- | --- |
| FailedOperation.TagAttachedResource | Tags associated with resources cannot be deleted. |
| ResourceNotFound.TagNonExist | Tag does not exist. |

## ModifyResourceTags

### 1. Interface Description

Interface request domain name: tag.api3.convergecloud.com.

This API is used to modify all tags associated with a resource.

Default interface request rate limit: 200 times per second.

Interface update time: 2024-12-06 17:02:47.

The interface is both signature verification and authentication.

### 2. Input Parameters

The following list of request parameters only lists interface request parameters and some public parameters, For a complete list of public parameters see public request parameters.

| Parameter name | Required | Allow NULL | Type | Description |
|---|---|---|---|---|
| Action | Yes | No | String | Public parameter, The value of this interface: ModifyResourceTags |
| Version | Yes | No | String | Public parameter, The value of this interface: 2018-08-13 |
| Region | No | No | String | Public parameter,Regional information This interface does not need to pass this parameter. |
| Resource | Yes | No | String | Resource description in 6-segment format.<br>Example: qcs:cvm:ap-region-x:uin/100004603715:instance/ins-qcmain |
| ReplaceTags | No | No | Array of Tag | Tags based on which you want to add bindings or modify bindings. If the resource specified for Resource is not bound to a tag key that you specify, the system adds a binding between the resource and the tag key. If the two are bound, the tag value corresponding to the tag key is changed to the tag value you specify. You must specify one of the following parameters: ReplaceTags and DeleteTags. You cannot specify an identical tag key for ReplaceTags and DeleteTags.<br>Example:View |
| DeleteTags | No | No | Array of TagKeyObject | Tag based on which you want to cancel bindings. You must specify one of the following parameters: ReplaceTags and DeleteTags. You cannot specify an identical tag key for ReplaceTags and DeleteTags.<br>Example:View |

### 3. Output Parameters

| Parameter name | Type | Description |
|---|---|---|
| RequestId | String | Unique request id, Each request will return. The RequestId of the request needs to be provided when locating the problem. |

### 4. Error Code

The following lists only the error codes related to the interface business logic, For other error codes, see public ErrorCode.

| Error Code | Description |
|---|---|
| InvalidParameterValue.ResourceDescriptionError | Invalid resource description. |
| LimitExceeded.TagKey | The number of created tag keys reaches the limit of 1,000. |
| LimitExceeded.TagValue | Each tag key can have up to 1,000 values. |
| InvalidParameter.Tag | Tag parameter error. |
| InvalidParameterValue.DeleteTagsParamError | DeleteTags cannot contain the elements in ReplaceTags or AddTags . |

# Data structure

## ServiceTypeFilter

Query resources.

Referenced by the following interfaces: DescribeResourcesBindTag

| name | required | NULL allowed | type | description |
|---|---|---|---|---|
| ServiceType | yes | no | String | Service Type<br>Example: "cvm" |
| ResourcePrefix | yes | no | Array of String | Resource Prefix<br>Example: ["instance"] |

## TagKeyObject

Tag key object.

Referenced by the following interfaces: ModifyResourceTags

| name | required | NULL allowed | type | description |
|---|---|---|---|---|
| TagKey | yes | no | String | Tag key<br>Example: TagKey |
| TagValue | no | no | String | Tag value<br>Example: TagValue |

## ResourceIdTag

Resource tag key value

Referenced by the following interfaces: DescribeResourceTagsByTagKeys, DescribeResourcesByTags

| name | required | NULL allowed | type | description |
|---|---|---|---|---|
| ResourceId | yes | no | String | Resource UID.<br>Example: 50 |
| TagKeyValues | yes | no | String | Tag key-value pair.<br>Example: TagValue_cvm |

## Tag

A tag key-value pair.

Referenced by the following interfaces: BatchCreateTag, ModifyResourceTags

| name | required | NULL allowed | type | description |
| --- | --- | --- | --- | --- |
| TagKey | yes | no | String | Tag key<br>Example: TagKey |
| TagValue | yes | no | String | Tag value<br>Example: TagValue |

# TagEntry

Tag entry

Referenced by the following interfaces: DescribeTagValues

| name | required | NULL allowed | type | description |
| --- | --- | --- | --- | --- |
| TagKey | no | no | String | Tag key<br>Example: TagKey_cvm |
| TagValue | no | no | String | Tag value<br>Example: TagValue |

# TagResource

Tag key-value pair and resource ID.

Referenced by the following interfaces: DescribeResourceTags, DescribeResourceTagsByResourceIds, DescribeResourcesBindTag

| name | required | NULL allowed | type | description |
| --- | --- | --- | --- | --- |
| TagKey | yes | no | String | Tag key<br>Example: TagKey |
| TagValue | yes | no | String | Tag value<br>Example: TagValue |
| ResourceId | yes | no | String | Resource ID.<br>Example: ResourceId |
| TagKeyMd5 | yes | no | String | MD5 value of a tag key.<br>Example: md5res |
| TagValueMd5 | yes | no | String | MD5 value of a tag value.<br>Example: md5res |
| ServiceType | yes | yes | String | Resource type.<br>Example: cos |

# TagWithDelete

A tag key-value pair and if deletion is allowed.

Referenced by the following interfaces: DescribeTags

| name | required | NULL allowed | type | description |
|---|---|---|---|---|
| TagKey | yes | no | String | Tag key<br>Example: TagKey1 |
| TagValue | yes | no | String | Tag value<br>Example: TagValue1 |
| CanDelete | yes | no | Uint64 | Indicates whether the tag can be deleted or not.<br>Example: TagKey1 |

# TagFilter

Tag filtering array. 'AND' relation if multiple arrays.

Referenced by the following interfaces: DescribeResourcesBindTag, DescribeResourcesByTags

| name | required | NULL allowed | type | description |
|---|---|---|---|---|
| TagKey | yes | no | String | Tag key<br>Example: TagKey |
| TagValue | no | no | Array of String | Tag value array. If the array includes multiple values, the relationship is OR.<br>Example: TagValue |

# ErrorCode

## Feature Description

If there is an Error field in the response, it means that the API call failed. For example:

```
{
    "Response": {
        "Error": {
            "Code": "AuthFailure.SignatureFailure",
            "Message": "The provided credentials could not be validated. Please check your signature is cor
rect."
        },
        "RequestId": "ed93f3cb-f35e-473f-b9f3-0d451b8b79c6"
    }
}
```

Code in Error indicates the error code, and Message indicates the specific information of the error.

## Error Code List

### Common Error Codes

| Error codes | Description |
| --- | --- |
| AuthFailure.InvalidSecretId | Invalid key (not a Cloud API key type). |
| AuthFailure.MFAFailure | MFA failed. |
| AuthFailure.SecretIdNotFound | Key does not exist. Check if the key has been deleted or disabled in the console, and if not, check if the key is correctly entered. Note that whitespaces should not exist before or after the key. |
| AuthFailure.SignatureExpire | Signature expired. Timestamp and server time cannot differ by more than five minutes. Please ensure your current local time matches the standard time. |
| AuthFailure.SignatureFailure | Invalid signature. Signature calculation error. Please ensure you've followed the signature calculation process described in the Signature API documentation. |
| AuthFailure.TokenFailure | Token error. |

| Error codes | Description |
| --- | --- |
| AuthFailure.UnauthorizedOperation | The request does not have CAM authorization. |
| DryRunOperation | DryRun Operation. It means that the request would have succeeded, but the DryRun parameter was used. |
| FailedOperation | Operation failed. |
| InternalError | Internal error. |
| InvalidAction | The API does not exist. |
| InvalidParameter | Incorrect parameter. |
| InvalidParameterValue | Invalid parameter value. |
| LimitExceeded | Quota limit exceeded. |
| MissingParameter | A parameter is missing. |
| NoSuchVersion | The API version does not exist. |
| RequestLimitExceeded | The number of requests exceeds the frequency limit. |
| ResourceInUse | Resource is in use. |
| ResourceInsufficient | Insufficient resource. |
| ResourceNotFound | The resource does not exist. |
| ResourceUnavailable | Resource is unavailable. |
| UnauthorizedOperation | Unauthorized operation. |
| UnknownParameter | Unknown parameter. |
| UnsupportedOperation | Unsupported operation. |
| UnsupportedProtocol | HTTPS request method error. Only GET and POST requests are supported. |
| UnsupportedRegion | API does not support the requested region. |

## Service Error Codes

| Error codes | Description |
| --- | --- |
| InvalidParameterValue.ServiceTypeInvalid | Service type error. |

| Error codes | Description |
| --- | --- |
| ResourceInUse.TagDuplicate | Tag already exists. |
| ResourceNotFound.TagNonExist | Tag does not exist. |
| FailedOperation.TagAttachedResource | Tags associated with resources cannot be deleted. |
| InvalidParameterValue.TagValueLengthExceeded | Tag value length exceeds limit. |
| InvalidParameterValue.TagKeyEmpty | Tag key cannot be empty. |
| InvalidParameterValue.DeleteTagsParamError | DeleteTags cannot contain the elements in ReplaceTags or AddTags . |
| InvalidParameterValue.TagValueCharacterIllegal | Tag value contains illegal characters. |
| ResourceNotFound.AttachedTagKeyNotFound | The tag key associated with the resource doesn't exist. |
| InvalidParameterValue.TagFilters | Values specified for TagFilters are invalid. |
| InvalidParameterValue.TagFiltersLengthExceeded | TagFilters does not exist or the value of TagFilters exceeds 6 characters. |
| InvalidParameterValue.RegionInvalid | Region error. |
| LimitExceeded.TagKey | The number of created tag keys reaches the limit of 1,000. |
| InvalidParameterValue.ResourceIdSizeInvalid | The number of resource exceeds the upper limit (50). |
| InvalidParameterValue.ReservedTagKey | Creation of system-reserved tag keys is not allowed. |
| LimitExceeded.TagValue | Each tag key can have up to 1,000 values. |
| InvalidParameter.Tag | Tag parameter error. |
| InvalidParameterValue.TagKeyCharacterIllegal | Tag key contains illegal characters. |
| InvalidParameterValue.ResourceDescriptionError | Invalid resource description. |
| InvalidParameterValue.UinInvalid | Uin parameter is invalid. |
| InvalidParameterValue.ResourcePrefixInvalid | Resource prefix error. |
| InvalidParameterValue.TagKeyLengthExceeded | Tag key length exceeds limit. |
| InvalidParameterValue.TagListEmpty | The TagList parameter is not specified. |