Copyright Notice

©2013-2019 Tencent Cloud. All rights reserved.

Copyright in this document is exclusively owned by Tencent Cloud. You must not reproduce, modify, copy or distribute in any way, in whole or in part, the contents of this document without Tencent Cloud's the prior written consent.

Trademark Notice

All trademarks associated with Tencent Cloud and its services are owned by Tencent Cloud Computing (Beijing) Company Limited and its affiliated companies. Trademarks of third parties referred to in this document are owned by their respective proprietors.

Service Statement

This document is intended to provide users with general information about Tencent Cloud's products and services only and does not form part of Tencent Cloud's terms and conditions. Tencent Cloud's products or services are subject to change. Specific products and services and the standards applicable to them are exclusively provided for in Tencent Cloud's applicable terms and conditions.
Contents

Developer Guide
Creating Request
Creating Request Overview
Bucket
Bucket Overview
Creating Buckets
Deleting Buckets
Object
Object Overview
Storage Class
Uploading Object
Simple Upload
Multipart Upload
Upload via Pre-Signed URL
Obtaining Object
Obtaining Object Simply
Download via Pre-Signed URL
Listing Object Keys
Copying Objects
Simple Copy
Multipart Copy
Deleting Object
Deleting a Single Object
Deleting Multiple Objects
Data Management
Lifecycle Management
Lifecycle Overview
Elements of Lifecycle Configuration
Configuring Lifecycle
Static Website Hosting
Inventory Overview
Bucket Tag Overview
Overview of object Tag
Event Notifications
Data Extraction
SELECT Overview
SELECT Command
SQL Functions
Reserved Words
Data Types
Operators
Log Management
Log Management Overview
Restrictions on Log Management
Storing Tencent Cloud Product Logs in COS
Data Disaster Recovery
Versioning
Versioning Overview
Using Versioning
  Versioning Configuration
Cross-region Replication
  Cross-region Replication Overview
  Replication Behavior Descriptions
  Cross-region Replication Configuration
Overview of Multi-AZ Feature
Data Security
  Server-side Encryption Overview
  Bucket Encryption Overview
Cloud Access Management (CAM)
  Access Control
    Basic Concepts of Access Control
    Notes on Principle of Least Privilege
    Policy-based Access Control
      Access Policy Language Overview
      Examples of Bucket Policies
    Access strategy evaluation process
    ACL-based Access Control
      ACL Overview
      Tag-based Access Control
        Manage project resources based on tags
Using CDN to Accelerate Access
  CDN Acceleration Overview
  CDN Acceleration Configuration
  Configuring GCD
Single link speed limit
Batch Processing
  Overview
  Performing Batch Operation
    Copying Objects in Batches
    Batch Resume archive object
  Managing Batch Operation Jobs
Monitoring and Alarming
Global Acceleration
  Overview
Data Processing
  Overview of picture processing
Developer Guide
Creating Request
Creating Request Overview

Last updated: 2019-07-22 14:30:37

Concept

Tencent Cloud COS is a web-based storage service accessed using the HTTP/HTTPS protocol. You can use the REST API or COS SDK to access COS.

Your COS access request must first pass the COS verification and authentication before COS starts to operate the resources. Therefore, depending on whether the identity is identifiable, COS access requests are divided into two types: anonymous requests and requests with signatures.

- Anonymous request: If the request does not include Authorization or related parameters, or the user identity cannot be identified based on the related characters, the request will be treated as an anonymous request for authentication.
- Request with signature: A request with a signature must contain the Authorization field in the HTTP header or the request package. The content of the field is generated based on Tencent Cloud security credentials (SecretID and SecretKey) and some eigenvalues of the request via an encryption algorithm.

To access COS using COS SDKs, you only need to configure your security credentials before initiating the request. To access COS using the REST API, calculate the request signature according to Request Signature.

Obtaining Security Credentials

Cloud Access Management (CAM) provides features and services related to accounts and credentials for COS, to help customers manage the permissions to access resources under their Tencent Cloud accounts in a secure way. You can use CAM to create, manage and terminate users (or user groups), and manage other users’ permissions to use Tencent Cloud resources through identity management and policy management.

Security certificates of the primary account

After logging in to the primary account, you can manage and obtain the security credentials (SecretID and SecretKey) of your primary account on the Cloud API Key page of CAM. The following is a key pair example:

36-character access key ID (SecretID): AKIDHZRLB9lbdp7Y7gyQq6BOk1997BGmUXg
32-character access Key (SecretKey): LYaWIuQmCSZ5ZMniUM6hiaLxHnW6XxRK

The access key can be used to identify the uniqueness of an account. After the signature is generated using the key and the request is sent, Tencent Cloud will identify the identity of the request initiator, and then perform verification and authentication for the identity, resources, operations, and conditions to determine whether to allow the operation.

The key of the primary account has all the operation permissions for all resources under the primary account. Disclosure of the key may cause loss of your cloud assets, so it is strongly recommended that you create sub-accounts and assign corresponding permissions for them, and then use the keys of sub-accounts to create requests for resource access and management.
Security certificates of sub-accounts

To manage users and cloud resources under your account in multiple dimensions, you can create multiple sub-accounts under your primary account to implement user-specific permission management. For more information on how to create a sub-account, see Sub-users in CAM.

Before using a sub-account to initiate an API request, you need to create a security credential for the sub-account, and then the sub-account will get a unique key pair, which can facilitate the identification of the identity. You can create user policies for different sub-accounts to control their access permissions to resources. You can also create user groups and associate one access policy to a user group to facilitate the central management of user grouping and resources.

With the corresponding permissions assigned, a sub-account can create or modify resources. The resources still belong to the primary account, and the resource cost will be deducted from the primary account.

Temporary security credentials

In addition to using security credentials of the primary account or sub-accounts to access resources, you can create roles and use the temporary security credentials of the roles to manage your Tencent Cloud resources. For more information on the role concept and how to use roles, see Role Overview.

As a virtual identity, a role does not have a permanent key. CAM provides a set of STS APIs used to generate temporary security credentials.

For more information on how to use the APIs and relevant examples, see Using Roles. See STS API to learn about how to generate temporary security credentials. Temporary security credentials contain only limited policies (operations, resources, and conditions), and are valid for a limited period (start and end time), so the generated temporary security credentials can be distributed or used directly.

You can call the API for to generate temporary security credentials and get a temporary key pair (tmpSecretId/tmpSecretKey) and a security token (sessionToken), which form the security credential that can be used to access COS. Here is a temporary security credential example:

41-character security token (SecurityToken): 5e776c4216ff4d31a7c74fe194a978a3ff2a42864
36-character temporary access key ID (SecretID): AKIDcAZnqgar9ByWq6m7ucIn8LNEuY2MkPCl
32-character temporary access key (SecretKey): VpxrX0IMCpHXWL0Wr3KQNcqljx1uhMqD

This API also returns the validity period of the temporary security credential via the expiration field, which means that this set of security credentials can only be used to initiate requests during this period.

Tencent Cloud COS provides a simple server SDK that can be used to generate temporary keys. You can visit COS STS SDK to obtain the SDK. To initiate the request using the REST API after getting the temporary security credential, you need to specify the value for the x-cos-security-token field in the HTTP header or the form-data of the POST request package to identify the security token used by the request, and then use the temporary access key pair to generate the request signature. For more information on how to initiate requests using the COS SDK, see the relevant sections in each SDK documentation.

Access Domain Name

REST API

The Region and Access Domain Name document provides a list of domain names that can be used to initiate access requests via the REST API.
It is recommended to use virtual hosting domain names to access COS buckets. When you initiate an HTTP request, the bucket to be accessed will be specified through the `Host` header, such as `<BucketName-APPID>.cos.<Region>.myqcloud.com`. Using virtual hosting domain names realizes the same feature as the "root" of a virtual server. Virtual hosting domain names can be used to host files such as favicon.ico, robots.txt, and crossdomain.xml, which are the content that many applications will retrieve from the "root" of the virtual server by default when identifying a hosted website.

You can also use a path request to access a bucket, such as `cos.<Region>.myqcloud.com/<BucketName-APPID>/`. The request `Host` and the signature must use `cos.<Region>.myqcloud.com`. SDK does not support this access method by default.

**Domain name of static websites**

If you enable the static website feature, a virtual hosting domain name will be assigned for you to use relevant features. Unlike the REST API, the domain name of static website only supports a few operations, such as GET/HEAD/OPTIONS Object, in addition to specific index pages, error pages and redirection configurations. Uploading or configuring resources is not supported.

The format of a domain name of a static website is `<BucketName-APPID>.cos-website.<Region>.myqcloud.com`. You can also log in to the console and go to the bucket's Basic Configuration - Static Website Configuration to get the domain name.

**COS Access via Private Network and Public Network**

The access domain names of COS adopt intelligent DNS resolution. For COS access via Internet (including different ISPs), we will detect and select the optimal linkage for you to access COS. If you have deployed a service in Tencent Cloud to access COS, the access in the same region will be automatically directed to the private network address. The cross-region access is not supported in the private network and the COS domain name is resolved to the public network address by default.

**How to determine an access via private network**

Tencent Cloud products within the same region access each other over a private network by default and no traffic fee is charged for these connections. For this reason, it is recommended to choose the same region when you purchase different Tencent Cloud products for cost saving.

The following shows how to determine whether it is an access via private network:

For example, when a CVM access COS, to determine whether a private network is used to access COS, use the `nslookup` command on the CVM to resolve the COS domain name. If a private network IP is returned, the access between CVM and COS is over a private network; otherwise, it is over a public network.

Generally, a private IP address takes the form of `10.*.*.*` or `100.*.*.*`, and a VPC IP address takes the form of `169.254.*.*`.

Assume that `mybucket-1250000000.cos.ap-guangzhou.myqcloud.com` is the address of target bucket, and the

```
nslookup mybucket-1250000000.cos.ap-guangzhou.myqcloud.com
```

below it indicates the access is over the private network.

```
Server: 10.138.224.65
Address: 10.138.224.65 #53

Name: mybucket-1250000000.cos.ap-guangzhou.myqcloud.com
Address: 10.148.214.13
Name: mybucket-1250000000.cos.ap-guangzhou.myqcloud.com
Address: 10.148.214.14
```

**Testing connectivity**

**Basic connectivity test**
COS uses the HTTP protocol to provide services. You can use the most basic tool `telnet` to test the connectivity to port 80 of the COS access domain.

An example of access through the public network:
```
telnet mybucket-1250000000.cos.ap-guangzhou.myqcloud.com 80
Trying 14.119.113.22...
Connected to gz.file.myqcloud.com.
Escape character is '^^]'.
```

An example of access through Tencent Cloud CVMs (basic network) within the same region:
```
telnet mybucket-1250000000.cos.ap-guangzhou.myqcloud.com 80
Trying 10.148.214.14...
Escape character is '^^]'.
```

An example of access through Tencent Cloud CVMs (VPC) within the same region:
```
telnet mybucket-1250000000.cos.ap-guangzhou.myqcloud.com 80
Trying 169.254.0.47....
Connected to 169.254.0.47.
Escape character is '^^]'.
```

Regardless of the access environment, if the command returns the `Escape character is '^^]'.` field, it indicates that the connection is successful.

**Test via the Internet**

Since the access to COS over the Internet involves the ISP network, which may prohibit you from testing connectivity using such tools as `ping` or `traceroute` of the ICMP protocol, so it is recommended to use the tools of the TCP protocol to test connectivity.

> The access via the Internet may involve multiple network environments. If the access is not smooth, check your local network linkage, or contact the local ISP.

If your ISP allows you to use the ICMP protocol, you can use the `ping`, `traceroute` or `mtr` tools to check your linkage. Otherwise, you can use the `psping` (Windows environment; download at the Microsoft official website) or such tools as `tcping` (cross-platform software) to test the latency.

**Test via the private network**

If you access the COS over the Tencent Cloud VPC in the same region, you may be unable to test connectivity using such tools as `ping` or `traceroute` of the ICMP protocol. It is recommended that you use the `telnet` command in the basic connectivity test to perform the testing.

You can also use the tools such as `psping` or `tcping` to test the latency to port 80 of the access domain. Before the test, make sure that the access domain name has been correctly resolved to the private network address using the `nslookup` command.
Bucket

Bucket Overview

Definition

A bucket is a carrier of objects, which can be considered as a "container" for storing objects. You can manage buckets and configure attributes of buckets in various methods such as the Tencent Cloud Console, APIs, and SDKs. For example, you can set a bucket for hosting a static website or set access permission to a bucket. For more information on bucket configuration, see the topics below:

- Creating a Bucket
- [Setting up a Static Website](https://intl.cloud.tencent.com/document/product/436/14984)
- Setting Access Permission
- Setting Hotlink Protection

Region

Region is where the COS IDC is located. COS allows users to create buckets in different regions. You can select the region closest to the location where you deploy your business for the buckets so as to reduce latency and cost, and meet the compliance requirements.

For example, if your business is distributed in South China, creating buckets in the Guangzhou region can accelerate object uploads and downloads. For more information on regions, see [Regions and Access Domain Names](https://intl.cloud.tencent.com/document/product/436/14984).

It is a must to specify a region when creating a bucket, and the specification is inalterable. All objects in the bucket are stored in the IDC in the region. You cannot set regions on the objects level.

Naming Conventions

A bucket name consists of two parts: **user-defined string** and **system-generated numeric string (APPID)**, which are connected by a dash ("-"). For example, in the bucket name `examplebucket-1250000000`, `examplebucket` is the user-defined string, and `1250000000` is the system-generated numeric string (APPID).

In the bucket name examples for APIs and SDKs, the naming format of a bucket is `<BucketName-APPID>`.

- System-generated numeric string (APPID): automatically assigned by the system, which you do not need to specify and is unique in Tencent Cloud.
- User-defined string: a string of characters entered manually by you, as specified below.

Naming convention for user-defined strings:

- You can only use lowercase letters [a-z], numbers [0-9], dashes "-", or a combination of them.
- The length of a user-defined string cannot exceed 50 characters.
- A bucket name cannot begin or end with "-".

The following are examples of valid bucket names:

- mybucket123-1250000000
- 1-newproject-1250000000
Types of Permission

A bucket provides two types of permissions by default: public and user.

**Public permissions**

Public permissions include "Private Read/Write", "Public Read/Private Write", and "Public Read/Write". You can modify bucket access permissions in **Permission Management** of the bucket in the COS Console. For more information, see Basic Concepts of Access Control.

- **Private Read/Write**
  Only the creator of the bucket and authorized accounts have Read/Write permission to the objects in the bucket. The default access permission of a bucket is Private Read/Write, which is recommended.

- **Public Read/Private Write**
  Anyone (including anonymous visitors) has Read permission to the objects in the bucket, but only the bucket creator and authorized accounts have Write permission to them.

- **Public Read/Write**
  Anyone (including anonymous visitors) has Read/Write permission to the objects in the bucket, which is not recommended.

**User permissions**

A root account has all the permissions (full control) for buckets by default. In addition, you can add sub-accounts that are granted permissions to read/write data and Read/Write permissions, and even **full access** to buckets.

This video illustrates bucket permissions in detail:

![Video](https:// intl.cloud.tencent.com/document/product/436/13324)

**Notes**

- COS stores objects using a flat structure instead of folders. For more information, see "Folders and Directories" in [Object Overview](https://intl.cloud.tencent.com/document/product/436/13324).
- You can create up to 200 buckets in total in all regions with one account. There is no limit on the number of objects in a bucket.
- In Tencent Cloud COS, the bucket name under one APPID must be unique.
- Once a bucket is created, it cannot be renamed. To rename a bucket, you need to delete it and create another one with the desired name.
- When creating a bucket, make sure to select the desired region, as the region cannot be changed once specified.
Creating Buckets

Last updated: 2019-11-13 15:08:21

Use Cases

To use COS, you need to create a bucket that helps you use and manage objects. You can create buckets via the console, APIs, or SDKs.

You can use the following sample code to create a bucket in the specified region. The supported parameters are:

- **Bucket**: specify your full bucket name in the format of `examplebucket-1250000000`.
- **Region**: select the region where your bucket resides. Once the bucket is created, the region cannot be modified. For more information, see [Regions and Access Domain Names](#) supported by COS.

Directions

**Via the COS Console**

You can create a bucket in the COS Console. For more information, see [Creating a Bucket](#) in Console Guide.

**Via REST API**

You can use the REST API directly to initiate a bucket creating request. For more information, see [PUT Bucket](#).

**Via the SDK**

You can directly call the bucket creating method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- [SDK for Android](#)
- [SDK for C](#)
- [SDK for C++](#)
- [SDK for .NET](#)
- [SDK for Go](#)
- [SDK for iOS](#)
- [SDK for Java](#)
- [SDK for Node.js](#)
- [SDK for PHP](#)
- [SDK for Python](#)
- [SDK for WeChat Mini Program](#)
Deleting Buckets

Use Cases

Buckets can be deleted via the console, APIs, or SDKs.

Only empty buckets can be deleted. If there are still objects in a bucket, its deletion will fail. Make sure that there are no objects in the bucket before deleting it.

Before deleting a bucket, make sure that the current identity has been granted the permission to delete it and that the correct Bucket and Region parameters are passed in.

Directions

Via the COS Console
You can delete a bucket in the COS Console. For more information, see Deleting a Bucket in Console Guide.

Via REST API
You can use the REST API directly to initiate a bucket deleting request. For more information, see DELETE Bucket in API documentation.

Via the SDK
You can directly call the bucket deleting method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- SDK for Android
- SDK for C
- SDK for C++
- SDK for .NET
- SDK for Go
- SDK for iOS
- SDK for Java
- SDK for JavaScript
- SDK for Node.js
- SDK for PHP
- SDK for Python
Object

Object Overview

Last updated: 2019-11-15 09:39:09

Definition

An object is the basic unit of COS and is stored in a bucket just like a photo stored in an album. You can manage objects in different ways including Tencent Cloud Console, APIs, and SDKs. An object is named in the format of `<ObjectKey>`.

Objects can be uploaded via simple upload or multipart upload.

- Use simple upload for objects less than 5 GB.
- Multipart upload is limited to no more than 10,000 parts (less than 5 GB per part) and a maximum object size of 48.82 TB.

Each object consists of an object key (ObjectKey), a data value (Value), and object metadata (Metadata).

- ObjectKey: the unique identifier of the object in the bucket.
- Value: the size of the uploaded object.
- Metadata: a set of name-value pairs that you can set when you upload an object.

You can configure objects in the console. For more information, see:

- Searching for Objects
- Viewing Object Information
- Setting Object Access Permission
- Setting Custom Headers

ObjectKey

Definition

An object in COS must contain a valid ObjectKey, which is the unique identifier of an object in a bucket.

For example, in an object's access address `examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com/folder/picture.jpg`, the ObjectKey is `folder/picture.jpg`.

Naming Convention

- You can use any UTF-8 character in a key name. However, for the maximum compatibility with other applications, it is recommended to use uppercase and lowercase letters and numbers (i.e., [a-z, A-Z, 0-9]), special characters ( ~, !, \, ., and * ), and a combination of them.
- The encoding length is up to 850 bytes.
- An object key does not support some ASCII control characters, including upward arrow (↑), downward arrow (↓), rightward arrow (→), and leftward arrow (←), corresponding to CAN (24), EM (25), SUB (26), and ESC (27).
- If the name of the uploaded file or folder contains Chinese characters, when you access or request the file or folder, the Chinese characters will be converted into a percent-encoded string according to URL-encoding rules.

For example, when you access `Document.doc`, the object key is `Document.doc`, while the percent-encoded string read is `%e6%96%87%e6%a1%a3.doc`.

The following are examples of valid key names:
Cloud Object Storage

- my-organization
- my.great_photos-2016/01/me.jpg
- videos/2016/birthday/video.wmv

Special Characters

Certain characters may need to be URL-encoded or referenced in the hexadecimal format. Some of them are non-printable characters and your browser might not be able to handle them. They’d require special handling, as shown below:

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
<th>URL-Encoded Value</th>
<th>Hexadecimal Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>,</td>
<td>:</td>
<td>@</td>
<td>21</td>
</tr>
<tr>
<td>&amp;</td>
<td>$</td>
<td>@</td>
<td>23</td>
</tr>
<tr>
<td>?</td>
<td>ASCII character ranges: 00-1F hexadecimal (0-31 decimal) and 7F (127 decimal)</td>
<td>(space)</td>
<td></td>
</tr>
<tr>
<td>`</td>
<td>^</td>
<td>`</td>
<td>60</td>
</tr>
<tr>
<td>{</td>
<td>}</td>
<td>[</td>
<td>59</td>
</tr>
<tr>
<td>~</td>
<td>%</td>
<td>#</td>
<td>35</td>
</tr>
<tr>
<td>&gt;</td>
<td>&lt;</td>
<td>ASCII 128-255 decimal</td>
<td></td>
</tr>
</tbody>
</table>

There are also some characters that require significant special handling to maintain consistency across all applications, so it is recommended to avoid them directly, as shown below:

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
<th>URL-Encoded Value</th>
<th>Hexadecimal Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>`</td>
<td>^</td>
<td>`</td>
<td>60</td>
</tr>
<tr>
<td>{</td>
<td>}</td>
<td>[</td>
<td>59</td>
</tr>
<tr>
<td>~</td>
<td>%</td>
<td>#</td>
<td>35</td>
</tr>
<tr>
<td>&gt;</td>
<td>&lt;</td>
<td>ASCII 128-255 decimal</td>
<td></td>
</tr>
</tbody>
</table>

Access Address

The access address of the object consists of a bucket access address and an object key, in the format of [bucket domain name]/[object key].

For example, when you upload the exampleobject.txt object to the examplebucket-1250000000 bucket in Guangzhou (South China), the access address for exampleobject.txt is examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com/exampleobject.txt.

Folder and Directory

As COS comes with no folders or directories, it will not create a project folder for uploading the object project/a.txt. To make it easier for you to get started, COS simulates the display mode of "folder" or "directory" in the console and graphical tools such as COS browser. This is implemented by creating an empty object with a key value of project/ and displaying it as a traditional folder.

For example, when you upload the object project/doc/a.txt via APIs or SDKs, the delimiter / simulates the display mode of "folder", and you can see the folders project and doc in the console. The folder doc is displayed under the folder project and contains the file a.txt.

Objects in the bucket are evenly distributed among distributed clusters. Therefore, you cannot directly get the size of all objects with a specified object key prefix. Instead, you can accumulate the size of each object to get the full size.

Deleting folders and directories is relatively complicated, as shown below:

<table>
<thead>
<tr>
<th>Deletion Means</th>
<th>Deletion</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Folder</td>
<td>project/a.txt</td>
</tr>
<tr>
<td></td>
<td></td>
<td>All objects with the object key prefix project/ will be deleted</td>
</tr>
<tr>
<td>Console</td>
<td>Object</td>
<td>project/doc/a.txt</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The project and doc folders will be retained</td>
</tr>
</tbody>
</table>
Deletion Means | Deletion | Result
--- | --- | ---
API and SDK | Object project/ or project/doc/ | The object project/doc/*.txt will be retained. If you want to delete the objects in the folder, use code traversal to delete them

**Metadata**

**Definition**

Metadata (aka HTTP header) is a set of name-value pairs in an object. It is the string sent by the server before the server sends HTML data using HTTP protocol to the browser. Modifying the HTTP header when uploading an object can alter page response forms or communicate configuration information, such as modifying caching time.

There are two types of metadata: system metadata and user-defined metadata.

> Modifying an object’s HTTP header does not modify the object itself.

**System Metadata**

This refers to the attribute information of the object, such as upload time or modification time.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>Current date and time</td>
</tr>
<tr>
<td>Content-Length</td>
<td>HTTP request content length in bytes as defined in RFC 2616, commonly used in API operations of PUT type</td>
</tr>
<tr>
<td>Last-Modified</td>
<td>Object creation date or the last modified date, whichever occurs later</td>
</tr>
<tr>
<td>Content-MD5</td>
<td>The Base64-encoded 128-bit MD5 checksum as defined in RFC 1864. This header is used to verify whether the file content has changed</td>
</tr>
<tr>
<td>Authorization</td>
<td>Authentication information, such as signature information used to verify the validity of request. For Public Read files, this header is not required</td>
</tr>
<tr>
<td>x-cos-version-id</td>
<td>Object version. If versioning is enabled for a bucket, the version ID of the object will be returned</td>
</tr>
<tr>
<td>ETag</td>
<td>Indicates the MD5 value of the uploaded file if the object is uploaded by PUT Object; indicates the unique ID of the uploaded file if the object is uploaded by multipart upload or using legacy of APIs, which cannot perform check though</td>
</tr>
<tr>
<td>Expect</td>
<td>HTTP request length in bytes as defined in RFC 2616</td>
</tr>
<tr>
<td>Connection</td>
<td>Connection status between the client and server. Enumerated values: keep-alive, close</td>
</tr>
</tbody>
</table>

**User-defined Metadata**

This refers to the object’s custom parameters, such as Content-Type, Cache-Control, Expires, and x-cos-meta-. For more information, see [Custom Object Headers](#).

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache-Control</td>
<td>The caching policy as defined in RFC 2616, which will be saved as the object’s metadata</td>
</tr>
</tbody>
</table>
## Cloud Object Storage

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Disposition/Encoding/Type</td>
<td>The file name/encoding format/content type (MIME) as defined in RFC 2616, which will be saved as the object's metadata</td>
</tr>
<tr>
<td>Expires</td>
<td>Object cache expiration time. For more information, see Expires Descriptions</td>
</tr>
<tr>
<td>x-cos-acl</td>
<td>Defines the ACL attribute of an object. Value range: private, public-read-write, public-read. Default value: private</td>
</tr>
<tr>
<td>x-cos-grant-*</td>
<td>Grants permission to the authorized user.</td>
</tr>
<tr>
<td>x-cos-meta-*</td>
<td>The header information allowed to be defined by users, which is returned as the object’s metadata. The size is limited to 2 KB</td>
</tr>
<tr>
<td>x-cos-storage-class</td>
<td>Sets the storage class of the object. Enumerated values: STANDARD, STANDARD_IA, ARCHIVE. Default value: STANDARD</td>
</tr>
<tr>
<td>x-cos-server-side-encryption</td>
<td>Indicates whether server-side encryption is enabled for the object. If you use a COS master key for encryption, enter AES256</td>
</tr>
</tbody>
</table>

## Object Sub-resources

COS has sub-resources that are associated with buckets and objects. Sub-resources belong to objects, so they do not exist independently; instead, they are always associated with other entities such as objects or buckets. An access control list (ACL) is the access control information list for a specific object, which is a sub-resource of a COS object.

An ACL contains an authorization list that identifies authorized users and the granted permissions to implement access control on the object. When you create an object, ACL identifies the object owner who can fully control the object. You can retrieve the object ACL or replace it with a new authorization list.

To update an ACL, you can only do so by replacing it.

## Access Permission Types

COS supports setting two types of permissions to objects: public permissions and user permissions.

**Public permissions**: Including inherited permission, Private Read/Write, and Public Read/Private write.

- Inherited permission: The object permissions inherited from the bucket is the same as the access permissions of the bucket itself. When you access an object with the “inherited bucket permission”, COS will match the bucket permission to respond to the access. A new object inherits the permission from its bucket by default.
- Private Read/Write: When you access an object with the Private Read/Write permission, the object can only be accessed with a request signature, regardless of the bucket permission.
- Public Read/Private Write: When you access an object with the Public Read permission, the object can be directly downloaded, regardless of the bucket permission.

**User permissions**: The root account has all the permissions of the object by default (i.e., full control). In COS, sub-accounts can be added to Read/Write data, Read/Write permissions, and have the full control permission.

**Use Cases**

Allow public access to a specified object in a Private Read/Write bucket or set a required authentication for a specific object in a Public Read/Write bucket.
COS offers three object storage classes for different object access frequencies: standard storage, standard infrequent access storage, and archive storage.

When an object is uploaded, if the storage class is not set, it defaults to **standard storage**.

### Standard Storage

Standard storage (Standard) provides object storage services featuring high reliability, availability, and performance.

It offers low access latency and high throughput, making it suitable for business scenarios where a large number of trending files is accessed in real time with frequent data interactions.

**Features**

- High performance and availability
- Data persistence: 99.999999999%
- Service availability: 99.95%
- Response: Within milliseconds
- Supported regions: All
- Storage cost: Standard

**Use Cases**

Trending videos, social networking photos, mobile apps, games, and dynamic websites.

### Standard Infrequent Access Storage

Standard infrequent access storage (Standard_IA) provides object storage services featuring high reliability and low storage cost and access latency.

It offers lowered pricing for storage and keeps the first-byte access time within milliseconds, ensuring that data can be fast retrieved with no wait required. However, data retrieval incurs fees. It is suitable for business scenarios where the access frequency is low (e.g., once or twice per month).

**Features**

- Low-frequency access, real-time response
- Data persistence: 99.999999999%
- Service availability: 99.9%
- Minimum billing duration: 30 days
- Minimum object size: 64 KB
- First-byte delay: Within milliseconds
- Supported regions: All
- Storage cost: Low

**Use Cases**

Online file storage, big data analytics, governmental/organizational business data, low-frequency archives, and monitoring data.
Archive Storage

Archive storage (Archive) provides objects storage services featuring high reliability, extremely low storage cost, and long-term retention.

It offers the lowest unit price of storage but requires a longer unfreezing time when data is read, making it suitable for archived data that needs to be retained for a long time.

Features

- High data persistence and service availability, low cost
- Data persistence: 99.999999999%
- Service availability: 99.9%
- Minimum billing duration: 90 days
- Minimum object size: 64 KB
- Response: Prior application for data restoration required
- Supported regions: Public cloud regions only
- Storage cost: Very low

Use Cases

Archive of compliance documents (e.g., archival data, medical images, and scientific materials), lifecycle files, operational logs, and remote disaster recovery.

Comparison of Storage Types

<table>
<thead>
<tr>
<th>Item</th>
<th>Standard</th>
<th>Standard_IA</th>
<th>Archive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data persistence</td>
<td>99.999999999%</td>
<td>99.999999999%</td>
<td>99.999999999%</td>
</tr>
<tr>
<td>Service availability</td>
<td>99.95%</td>
<td>99.9%</td>
<td>99.9%</td>
</tr>
<tr>
<td>Response</td>
<td>Within milliseconds</td>
<td>Within milliseconds</td>
<td>Prior application for data restoration required</td>
</tr>
<tr>
<td>Minimum object size</td>
<td>Calculated by actual object size</td>
<td>64 KB</td>
<td>64 KB</td>
</tr>
<tr>
<td>Minimum billing duration</td>
<td>No limit</td>
<td>30 days</td>
<td>90 days</td>
</tr>
<tr>
<td>Supported regions</td>
<td>All</td>
<td>All</td>
<td>Public cloud regions only</td>
</tr>
<tr>
<td>Storage cost</td>
<td>Standard</td>
<td>Low</td>
<td>Very low</td>
</tr>
<tr>
<td>Data retrieval cost</td>
<td>None</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Read/write request cost</td>
<td>Very low</td>
<td>Low</td>
<td>Very low (data needs to be restored to standard storage)</td>
</tr>
</tbody>
</table>
Uploading Object
Simple Upload

Use Cases

This operation is suitable for uploading an object smaller than 5 GB in a single request. For objects larger than 5 GB, you must use the multipart upload method.

If your object is large (for example, 100 MB), it is recommended that you use multipart upload in a high-bandwidth or weak network environment.

Directions

Via the COS Console
You can upload an object in the COS Console. For more information, see Uploading an Object in Console Guide.

Via REST API
You can use the REST API directly to initiate a simple object upload request. For more information, see PUT Object.

Via the SDK
You can directly call the simple object upload method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- SDK for Android
- SDK for C
- SDK for C++
- SDK for .NET
- SDK for Go
- SDK for iOS
- SDK for Java
- SDK for JavaScript
- SDK for Node.js
- SDK for PHP
- SDK for Python
Use Cases

Multipart upload is well suited for uploading large objects in weak network or high-bandwidth environments. The COS Console and SDKs can break a single object into multiple parts and then upload them. You can also do so by yourself and upload each part through API calls. Multipart upload has the following benefits:

- In a weak network, smaller part size minimizes the impact of restarting a failed upload due to network errors.
- In a high-bandwidth environment, multipart upload can maximize the use of your available bandwidth by uploading object parts in parallel. Uploading parts out of order does not affect the final merged object.
- With multipart upload, you can pause and resume the upload of a single large object at any time. All incomplete multipart uploads can be resumed unless aborted.
- Multipart upload can also be used to begin an upload before you know the final object size. You can initiate an upload and then merge the parts to get the full object size.

When uploaded, the parts are numbered consecutively. You can upload each part separately or upload them in any order. COS will merge the parts into an object based on their numbers. If the upload of any part fails, the failed part can be uploaded again without affecting other parts and the content as a whole. In a weak network environment, multipart upload is recommended for objects larger than 20 MB. In a high-bandwidth environment, multipart upload is recommended for objects larger than 100 MB.

Directions

Via REST API

You can use the REST API directly to initiate a multipart upload request. For more information, see the following API documents:

- Initiate Multipart Upload
- Upload Part
- Complete Multipart Upload
- Abort Multipart Upload

Via the SDK

You can directly call the multipart upload method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- SDK for Android
- SDK for C
- SDK for C++
- SDK for .NET
- SDK for Go
- SDK for iOS
- SDK for Java
- SDK for JavaScript
- SDK for Node.js
- SDK for PHP
- SDK for Python
Upload via Pre-Signed URL

Last updated: 2019-11-13 15:17:07

Use Cases

All buckets and objects are private by default. If you want a third party to be able to upload an object to your bucket but don't want them to use CAM account or temporary keys, signatures can be provided by pre-signed URLs for temporary upload operations. Anyone who receives a valid pre-signed URL can upload an object.

When creating a pre-signed URL, you can include an object key in your signature so that the object can only be uploaded to the specified object key. Besides, the validity period of pre-signed URLs can be provided in SDKs to ensure that expired URLs will not be used by any unauthorized party.

Directions

Via the SDK

You can call the pre-signed URL method in the SDK directly. For more information, see the SDK documentation for the corresponding programming language below:

- SDK for Android
- SDK for C
- SDK for C++
- SDK for .NET
- SDK for Go
- SDK for iOS
- SDK for Java
- SDK for JavaScript
- SDK for Node.js
- SDK for PHP
- SDK for Python
Obtaining Object

Obtaining Object Simply

Use Cases

You can initiate a request to obtain objects directly in COS. You have the following options when obtaining an object:

- Obtain a complete object: Obtain the complete object data directly by initiating a GET request.
- Obtain a part of an object: Use the Range request header in a GET request to retrieve a specific range of bytes of an object. Retrieving multiple ranges is not supported.

The object’s metadata will be returned along with the object’s content as an HTTP response header. The GET request supports overwriting certain metadata values in the response using URL parameters. For example, the response value of Content-Disposition can be overwritten. Response headers that support modification include:

- Content-Type
- Content-Language
- Expires
- Cache-Control
- Content-Disposition
- Content-Encoding

Directions

Via the COS Console

You can obtain an object in the COS Console. For more information, see Downloading Objects in Console Guide.

Via REST API

You can use the REST API directly to initiate an object obtaining request. For more information, see GET Object.

Via the SDK

You can directly call the object download method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- SDK for Android
- SDK for C
- SDK for C++
- SDK for .NET
- SDK for Go
- SDK for iOS
- SDK for Java
- SDK for JavaScript
- SDK for Node.js
- SDK for PHP
- SDK for Python
Download via Pre-Signed URL


Use Cases

All buckets and objects are private by default. If you want any third party to be able to download an object without using CAM account or temporary keys, provide the third parties with signatures via pre-signed URLs for download operations. Anyone who receives a valid pre-signed URL can download an object.

When creating a pre-signed URL, you can include object keys in your signature to specify the objects allowed for download. Besides, the validity period of pre-signed URLs can be provided in SDKs to ensure that expired URLs will not be used by any unauthorized party.

Directions

Via the SDK

You can call the pre-signed URL method in the SDK directly. For more information, see the SDK documentations for the corresponding programming language below:

- SDK for Android
- SDK for C++
- SDK for .NET
- SDK for Go
- SDK for iOS
- SDK for Java
- SDK for JavaScript
- SDK for Node.js
- SDK for PHP
- SDK for Python
Use Cases

Tencent Cloud COS supports listing keys by prefix. You can use the separator ( `/` ) in a key to implement a hierarchical structure similar to the traditional file system. In COS, you can use separators to select and browse keys hierarchically.

You can list all keys in a single bucket in UTF-8 binary order of prefixes or filter the key list by specifying the prefix. For example, adding the parameter `t`, would list the `tencent` object, while skipping objects prefixed with `a` or other characters.

Keys can be reorganized based on the added separator ( `/` ). You can use the prefix and separator together to implement a folder retrieval feature. For example, if you add the prefix parameter `t` and the separator ( `/` ), eligible keys such as `tencent/cos` will be listed.

COS supports storing an unlimited number of objects in a single bucket, so the key list may be very large. For management purposes, a maximum of 1,000 key values are returned for one List Objects request, and a marker will be returned to indicate whether the result is truncated. You can send a series of List Objects requests based on markers and separators to list all key values or search for the desired content.

Directions

Via REST API
You can use the REST API directly to initiate an object key listing request. For more information, see GET Bucket.

Via the SDK
You can directly call the object list querying method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- SDK for Android
- SDK for C
- SDK for C++
- SDK for .NET
- SDK for Go
- SDK for iOS
- SDK for Java
- SDK for JavaScript
- SDK Node.js
- SDK for PHP
- SDK for Python
- SDK for WeChat Mini Program
Use Cases

The copy operation creates a copy of an object that is already stored in COS. You can create a copy of your object up to 5 GB in a single operation. To copy an object over 5 GB, you must use the multipart upload API. With the copy operation, you can:

- Create a copy of an object.
- Rename the object by copying it and deleting the original one.
- Modify the storage class of the object. In the copy operation, you can select the same object key as both the source and destination and modify the storage class.
- Copy objects across different COS regions.
- Modify object metadata. In the copy operation, you can select the same object key as both the source and destination and modify object metadata.

In the copy operation, the metadata of the original object is inherited by default, but the creation date is subject to the new object.

- Copy and paste is not supported for objects in the archive storage class.
- The standard storage (multi-AZ) class only supports replication into the exactly same class rather than standard storage, low frequency, or archive storage class.

Directions

Configuring via the COS Console

You can copy an object in the COS Console. For more information, see Copying Objects in Console Guide.

Configuring via REST API

You can use the REST API directly to initiate an object copying request. For more information, see PUT Object - Copy.

Via the SDK

You can directly call the object copying method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- SDK for Android
- SDK for C
- SDK for C++
- .NET SDK
- SDK for Go
- SDK for iOS
- SDK for Java
- SDK for JavaScript
- Node.js SDK
- SDK for PHP
- SDK for Python
- SDK for WeChat Mini Program
Use Cases

For copying an object greater than 5 GB, you must use the multipart copy. Use the multipart upload API to create an object, and use the Upload Part - Copy API to carry the x-cos-copy-source header to specify the source object. The process is as follows:

1. Initialize an object for multipart upload.
2. Copy the data of the source object. You can specify the x-cos-copy-source-range header and copy up to 5 GB of data at a time.
3. Complete the multipart upload.

SDKs provided by COS can be used to easily implement multipart copy.

Directions

Via REST API

You can use the REST API directly to initiate a multipart copy request. For more information, see the following API documentations:

- Initiate Multipart Upload
- Upload Part - Copy
- Complete Multipart Upload
- Abort Multipart Upload

Via the SDK

You can call the multipart copy method in the SDK directly. For more information, see the SDK documentation for the corresponding programming language below:

- SDK for Android
- SDK for C++
- SDK for iOS
- SDK for Java
- SDK for JavaScript
- SDK for Node.js
- SDK for PHP
- SDK for Python
Deleting Object
Deleting a Single Object

Use Cases

COS supports deleting one or multiple objects directly. To delete one object, you only need to provide its object key and call an API request to delete it.

Directions

Via the COS Console
You can delete an object in the COS Console. For more information, see Deleting an Object in Console Guide.

Via REST API
You can use the REST API directly to initiate a single object deleting request. For more information, see DELETE Object.

Via the SDK
You can directly call the single object deleting method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- SDK for Android
- SDK for C
- SDK for C++
- SDK for .NET
- SDK for Go
- SDK for iOS
- SDK for Java
- SDK for JavaScript
- SDK for Node.js
- SDK for PHP
- SDK for Python
- SDK for WeChat Mini Program
Deleting Multiple Objects

Use Cases

COS supports batch deletion of multiple objects. You can delete objects in batches via the console, APIs, and SDKs.

By default, when the deletion task is completed, a null will be returned. If an error occurs, an error message will be returned.

A maximum of 1,000 objects can be deleted in a single request. To delete more objects, split the list and send the request separately.

Directions

Via the COS Console
You can delete multiple objects in batches in the COS Console. For more information, see Deleting an Object in Console Guide.

Via REST API
You can use the REST API directly to initiate a multiple objects deleting request. For more information, see DELETE Multiple Objects.

Via the SDK
You can directly call the multiple objects deleting method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- SDK for Android
- SDK for C
- SDK for C++
- SDK for Go
- SDK for iOS
- SDK for JavaScript
- SDK for Node.js
- SDK for PHP
- SDK for Python
Data Management
Lifecycle Management
Lifecycle Overview

COS supports object-based lifecycle configuration. By sending specific description language to a bucket, COS can automatically perform predefined actions on objects to which a rule is applied.

A lifecycle can be set to as long as 3,650 days.

Use Cases

Log Record
With corresponding lifecycle configuration, log data stored in COS can be automatically archived after 30 days or deleted after 2 years.

Data Transition
Hot data is frequently accessed in a short period of time after the upload, but then is less or not accessed at all after some time. Therefore, you may set the lifecycle rules to store the data uploaded 30 days ago in Standard_IA storage class and store the data uploaded 60 days ago in Archive storage class. This process is called data transition.

Archive Management
When you use COS for file archive management, you need to save all historical versions of files for a long time in accordance with compliance requirements in financial, medical, and other industries. In this case, you can configure the lifecycle to transition and store the historical versions of files in Archive storage class.

Support Descriptions

Supported Actions
- Data transition: transition objects to Standard_IA or Archive storage class after a specified period of time.
- Deletion after expiration: automatically delete objects after their specified expiration time.

Supported Resources
- Specified based on prefixes: objects with different prefixes are processed according to applicable rules.
- Versioning-enabled objects: you can manage the current and historical versions of objects separately as needed.
- Objects with delete markers: when all old versions of objects are cleared, you can specify to remove their delete markers.
- Incomplete multipart uploads: process incomplete multipart upload tasks.

Supported Time Conditions
- Based on the number of days: specify in how many days after an object is last modified the corresponding action will be performed.
- Based on a specific date: specify a date when the corresponding action will be performed.

Notes
Data Transition

Supported Regions
Data transition to the Archive storage class is supported in public cloud regions.

One-way Transition
Data transition is one-way (from Standard storage to Standard_IA storage to Archive storage or from Standard storage to Archive storage) and cannot be in the reversed way. You can only call PUT Object - Copy (for non-archive storage classes) or POST Object restore (for Archive storage class only) to restore data from a colder storage class to a hotter class.

Eventual Consistency
If multiple conflicting rules (excluding deletion after expiration) are configured for the same set of objects, these rules will be performed in a chronological order. The objects will be transitioned to the coldest storage class according to the rules.

Deletion after Expiration

How it Works
When an object matches the specified lifecycle rule of deletion after expiration, COS will add it to the async deletion queue. There will be a delay in the actual deletion. You can get the current status of the object by performing a GET or HEAD Object operation.

Eventual Consistency
If multiple conflicting rules are configured for the same set of objects, these rules will be performed in chronological order in accordance with expiration time. Deletion after expiration takes precedence over before transition from one storage class to another.

Cost Considerations

Execution Instructions
Once triggered, actions configured at any time will always be executed at 00:00 Beijing time (GMT+8) the next day in COS. As objects need to be added to an async queue before execution, for objects uploaded after the configuration time that are matched by a rule, actions will be performed on them before 00:00 the next day.

When any accident occurs or there are too many objects in the bucket, lifecycle execution may fail. For failures due to other reasons, execute GET or HEAD Object to get the current object status.

Tencent Cloud cannot provide an accurate bill unless the lifecycle execution is completed.

Time-insensitivity
Please note that the minimum storage duration in Standard_IA or Archive storage class is 30 or 60 days, respectively; otherwise, additional storage fees will be incurred upon data transition or deletion. COS will not check lifecycle configurations less than 30/60 days; therefore, it will execute correct configurations upon your request.

For example, if an object in Standard_IA storage class is transitioned before it is stored for 30 days, it will start incurring Archive storage fees on the transition day and continue incurring Standard_IA storage fees until the 30th day.
Another example is that if an archived object is deleted upon expiration before it is stored for 60 days, it will continue incurring Archive storage fees until the 60th day.

**Size-insensitivity**

There are minimum object size requirements in both the Standard_IA and Archive storage classes. For example, if an object below 64 KB is uploaded to the Standard_IA storage class, it will be calculated as 64 KB. COS will not check the file size; instead, it will perform object conversion operations unconditionally according to the specified rule.
Elements of Lifecycle Configuration

Last updated: 2019-06-14 18:46:43

Basic Structure

A lifecycle configuration is specified as XML and consists of one or more lifecycle rules. The structure is as follows:

```xml
<LifecycleConfiguration>
  <Rule>
    <ID>your lifecycle name</ID>
    <Status>Enabled</Status>
    <Filter>
      <And>
        <Prefix>projectA/</Prefix>
        <Tag>
          <Key>key1</Key>
          <Value>value1</Value>
        </Tag>
        <And>
          <Filter>**transition/expiration actions**
        </Rule>
        ...
      </Filter>
    </Filter>
  </Rule>
</LifecycleConfiguration>
```

Each rule contains the following:

- **ID**: (optional) Indicates the content of the rule and is customizable.
- **Status**: Indicates whether the rule is enabled or disabled.
- **Filter**: Identifies objects to which the rule applies.
- **Action**: Actions that need to be performed on objects that match the above description.
- **Time**: Days (calculated from the date when an object is last modified) or Date based on which actions are performed on objects.

Rule Description

**Filter element**

**For all objects in a bucket**

You can specify an empty filter, in which case the rule applies to all objects in the bucket.

```xml
<LifecycleConfiguration>
  <Rule>
    <Filter />
    <Status>Enabled</Status>
    **transition/expiration actions**
  </Rule>
</LifecycleConfiguration>
```

**Based on object key prefixes**

Specify a prefix-based filter so that a rule applies only to objects with specific prefixes. For example, you can filter out objects based on the prefix `logs/`:
Based on object tags
Specify a filter based on key and value so that a rule applies only to objects with the specific tag. For example, you can filter out objects based on tags key=type and value=image:

```xml
<LifecycleConfiguration>
  <Rule>
    <Filter>
      <Prefix>logs/</Prefix>
      <Tag>
        <Key>type</Key>
        <Value>image</Value>
      </Tag>
      <Filter>
        <Status>Enabled</Status>
      **transition/expiration actions**
    </Filter>
  </Rule>
</LifecycleConfiguration>
```

Use multiple filters
In Tencent Cloud COS, you can combine multiple filters using AND. For example, you can filter out objects based on the prefix logs/ as well as tags key=type and value=image:

```xml
<LifecycleConfiguration>
  <Rule>
    <Filter>
      <And>
        <Prefix>logs/</Prefix>
        <Tag>
          <Key>type</Key>
          <Value>image</Value>
        </Tag>
        <Filter>
          <Status>Enabled</Status>
        **transition/expiration actions**
      </And>
    </Filter>
  </Rule>
</LifecycleConfiguration>
```

Elements to describe actions
You can specify one or more predefined actions in a lifecycle rule so that these actions are performed when any objects meet the rule.

Transition action
Specify the Transition action to transition objects from one storage class to another. For a versioned bucket, the Transition action applies to the current object version. You can set the transition date to as short as 0 day. For example, transition objects to Archive Storage after 30 days:
<Transition>
<StorageClass>ARCHIVE</StorageClass>
<Days>30</Days>
</Transition>

Deletion after expiration
Specify the Expiration action to delete expired objects. For a non-versioned bucket, expired objects will be permanently deleted. For a versioned bucket, expired objects are added with DeleteMarker and become the current version. For example, delete objects with an expiration of 30 days:

<Expiration>
<Days>30</Days>
</Expiration>

Incomplete Multipart Upload
Specify the AbortIncompleteMultipartUpload action to delete multipart upload tasks with specific UploadId if they are not successfully completed within the predefined time period. In this case, these tasks also cannot be resumed or indexed. For example, abort multipart upload tasks not successfully completed within 7 days:

<AbortIncompleteMultipartUpload>
<Days>7</Days>
</AbortIncompleteMultipartUpload>

Objects of non-current versions
In a versioned bucket, Transition action only applies to objects of the latest version and Expiration action only results in adding deletion markers to the objects. Therefore, Tencent Cloud COS provides the following actions to objects of non-current version:

Specify the NoncurrentVersionTransition action to transition the current version of objects to another storage class at a specified time. For example, transition the non-current versions of objects to Archive Storage after 30 days:

<NoncurrentVersionTransition>
<StorageClass>ARCHIVE</StorageClass>
<Days>30</Days>
</NoncurrentVersionTransition>

Specify the NoncurrentVersionExpiration action to delete the current version of objects expired at a specified time. For example, delete the non-current versions of objects expired after 30 days:

<NoncurrentVersionExpiration>
<Days>30</Days>
</NoncurrentVersionExpiration>

When all non-current object key versions are deleted and the current object version is DeleteMarker, specify the ExpiredObjectDeleteMarker action to delete the DeleteMarker. For example, delete the DeleteMarker of the expired objects after 31 days:

<ExpiredObjectDeleteMarker>
<Days>31</Days>
</ExpiredObjectDeleteMarker>

Time element
Based on number of days
Days refers to the number of days since an object was last modified. For example, if an object is set to be transitioned to Archive Storage after 0 day, when the object is uploaded at 2018-01-01 23:55:00 GMT+8, it will be added to the transition queue at 2018-01-02 00:00:00 GMT+8 and transitioned before 2018-01-02 23:59:59 GMT+8.

For example, if an object is set to be deleted after 1 day due to expiration, when the object is uploaded at 2018-01-01 23:55:00 GMT+8, it will be added to the expiration queue at 2018-01-03 00:00:00 GMT+8 and deleted before 2018-01-03 23:59:59 GMT+8.

**Based on a specific date**

Perform the predefined action to all qualified objects based on the filter criteria at the specific date. The date value must conform to the ISO 8601 format. The time is always 00:00 GMT+8.

For example, use **2018-01-01T00:00:00+08:00** for January 1, 2018.
Configuring Lifecycle

Use Cases

With lifecycle configuration, certain predefined actions can be automatically performed when a rule is applied to objects. For example:

- Transition: transition objects to STANDARD_IA storage or ARCHIVE storage after a specified time of period.
- Expiration: automatically delete objects after their specified expiration time.

For more information, see Lifecycle Overview and Elements of Lifecycle Configuration.

Directions

Via the COS Console

You can configure the lifecycle in the COS Console. For more information, see Lifecycle Management in Console Guide.

Via REST API

You can configure and manage the lifecycle of objects in a bucket through the REST API as described in the following API documentations:

- PUT Bucket lifecycle
- GET Bucket lifecycle
- DELETE Bucket lifecycle

Via the SDK

You can directly call the lifecycle method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- SDK for Android
- SDK for C
- SDK for C++
- SDK for .NET
- SDK for Go
- SDK for iOS
- SDK for Java
- SDK for JavaScript
- SDK for Node.js
- SDK for PHP
- SDK for Python
Static Website Hosting

Last updated: 2020-04-23 17:22:37

Concepts

A static website is a website that contains static content (such as HTML) or client scripts. Users can configure static websites for buckets with a custom domain name through the console. A dynamic website contains server scripts such as PHP, JSP, or ASP.NET, and needs to be processed on a server. You can host static websites on Tencent Cloud COS, but cannot write server scripts. When deploying a dynamic website, you are recommended to use a CVM for server code deployment.

Samples

A user created a bucket named examplebucket-1250000000 and uploaded the following file:

- index.html
- 404.html
- 403.html
- test.html
- docs/a.html
- images/

Static website

**Before enabled:** Access the bucket via the following default domain name. When a download prompt pops up, save the `index.html` file to the local.


**After enabled:** Access the bucket via the following access node, and then you can view the content of `index.html` directly in the browser.


Forced HTTPS

**Before enabled:** When the request source is HTTP, the access node URL keeps the HTTP unencrypted transport protocol:

http://examplebucket-1250000000.cos-website.ap-guangzhou.myqcloud.com

**After enabled:** The access node always maintains the HTTPS encrypted transport protocol regardless of whether the request source is HTTP or HTTPS:

https://examplebucket-1250000000.cos-website.ap-guangzhou.myqcloud.com

Index document

An index file, the homepage of the static website, is a page returned when the root directory or any subdirectory of a website is requested, and is usually named `index.html`. When you access a static website via a bucket access domain name, such as `https://examplebucketbucket-1250000000.cos-website.ap-guangzhou.myqcloud.com`, and no specific page is requested, the Web server will return the homepage.

When your user accesses any directory (including the root directory) in a bucket using a URL ended with `/`, the index document in that directory is matched automatically. `/` is not mandatory in the root URL, so the index document is returned in response to either of the following URLs.
Cloud Object Storage

http://www.examplebucket.com/
http://www.examplebucket.com

If folders are created in the bucket, the index document needs to be added at each folder level.

Error document

Suppose that when you visit the following page before configuring the error document, a 404 status code is returned, and the default error information displays in the page.

https://examplebucket-1250000000.cos-website.ap-guangzhou.myqcloud.com/webpage.html

When you visit the following page after configuring the error document, a 404 status code is also returned, but the specific error information displays in the page.

https://examplebucket-1250000000.cos-website.ap-guangzhou.myqcloud.com/webpage.html

Redirection rules

To configure redirection rules for a hosted static website, the path of replacing file must be an object path in your bucket.

Configure error code redirection

If the webpage.html document is set to Private Read/Write, when a user tries to access it, a 403 error is returned. After the 403 error code is redirected to 403.htm, the browser will return the content of 403.html.

If you do not configure a 403.html document, the browser will return an error document or default error message.

<table>
<thead>
<tr>
<th>Redirect rules</th>
<th>Type</th>
<th>Description</th>
<th>Force HTTPS</th>
<th>Rule</th>
<th>Replace content</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>http error code</td>
<td>403</td>
<td>Replace path</td>
<td></td>
<td>403.html</td>
<td></td>
<td>Delete</td>
</tr>
</tbody>
</table>

Configure prefix match

1. When you rename a folder from docs/ to documents/, the user will get an error when accessing the docs/ folder. So, you can redirect the request with the prefix docs/ to documents/.

<table>
<thead>
<tr>
<th>Redirect rules</th>
<th>Type</th>
<th>Description</th>
<th>Force HTTPS</th>
<th>Rule</th>
<th>Replace content</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefix matching</td>
<td>docs/</td>
<td>Replace prefix</td>
<td></td>
<td>documents/</td>
<td></td>
<td>Delete</td>
</tr>
</tbody>
</table>

2. When you delete the images/ folder (i.e., deleting all objects with the prefix images/), you can add a redirection rule to redirect requests for any object with the prefix images/ to test.html.

©2013-2019 Tencent Cloud. All rights reserved. Page 38 of 151
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Force HTTPS</th>
<th>Rule</th>
<th>Replace content</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefix matching</td>
<td>images/</td>
<td>☑</td>
<td>Replace prefix</td>
<td>test.html</td>
<td>Delete</td>
</tr>
</tbody>
</table>

Add Rules
Inventory Overview

Last updated : 2019-12-17 12:42:00

What Is Inventory

Inventory is a feature that helps users manage bucket objects. It operates periodically and as an alternative to the COS synchronous List API operation. COS can daily or weekly scan a specified object or objects with a same prefix in a bucket based on an inventory task configured by the user, and export and store an inventory report as a CSV file in the bucket specified by the user. The file lists stored objects and corresponding metadata, and records the desired object attribute information based on the configuration.

You can use the inventory feature for various purposes, including but not limited to:

- Review and report replication and encryption states of an object.
- Streamline and speed up service workflows and big data jobs.

- You can configure multiple inventory tasks for one bucket. COS doesn’t directly read the content of an object, but only scan the attribute information such as metadata of the object.
- Currently, the inventory feature is not available for COS buckets in financial cloud regions and outside China regions.

Inventory Parameters

After you configure an inventory task for a bucket, COS regularly scans a specified object in the bucket based on your configuration, and exports an inventory report in .csv format. Currently, the inventory report can record the following information:

<table>
<thead>
<tr>
<th>Inventory Information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppID</td>
<td>Account ID</td>
</tr>
<tr>
<td>Bucket</td>
<td>Name of the bucket for which an inventory task is executed.</td>
</tr>
<tr>
<td>Key</td>
<td>Name of an object file in a bucket. When using the CSV file format, the key name is URL-encoded and must be decoded before you can use it.</td>
</tr>
<tr>
<td>VersionId</td>
<td>Version ID of an object. After version control is enabled for a bucket, COS specifies a version ID for the object added to the bucket. If the list is for the current version only, the inventory does not include this field.</td>
</tr>
<tr>
<td>IsLatest</td>
<td>Sets to True if the latest object version is used. This field is not included if the list is only for the current version of objects.</td>
</tr>
<tr>
<td>IsDeleteMarker</td>
<td>Sets to True if the object is a delete marker. This field is not included if the list is only for the current version of objects.</td>
</tr>
<tr>
<td>Size</td>
<td>Size of an object (in bytes)</td>
</tr>
<tr>
<td>LastModifiedDate</td>
<td>Last modified date of an object (the most recent date prevails)</td>
</tr>
<tr>
<td>ETag</td>
<td>Hash value of an object. It displays only modification to the content of an object, rather than to its metadata. The ETag may be or may not be the MD5 checksum of the data of the object, and this depends on how the object is created and encrypted.</td>
</tr>
<tr>
<td>StorageClass</td>
<td>Storage class of the object. For more information, see Storage Class</td>
</tr>
<tr>
<td>Inventory Information</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>IsMultipartUploaded</td>
<td>Sets to True if the object is uploaded in multiple parts. For more information, see Multipart Upload</td>
</tr>
<tr>
<td>Replicationstatus</td>
<td>Sets to PENDING, COMPLETED, FAILED, or REPLICA. For more information, see Replication Behavior Descriptions.</td>
</tr>
</tbody>
</table>

## How to Configure the Inventory

Before configuring the inventory, you need to understand two concepts:

- **Source bucket**: The bucket for which the inventory is to be enabled
  - Includes the objects listed in the inventory
  - Includes the configurations of the inventory

- **Destination bucket**: The bucket where the inventory list file is stored
  - Includes files in the inventory list
  - Includes the manifest files describing the location of the inventory list

To configure the inventory, do the following steps:

### Specifying Information about Objects to Be Analyzed in the Source Bucket

To inform COS of object information to be analyzed, you need to configure the following information in the source bucket for the inventory:

- **Select object versions**: List all the versions of the object or list only the current version. If you select to list all the versions of the object, COS adds information about the object in all historical versions to the inventory report. If you select to list only the current version, COS records only the object in the latest version.

- **Specify the object attributes to be analyzed**: You need to inform COS of object attributes that need to be recorded in the inventory report. Currently, the following object attributes are supported: account ID, name of the source bucket, name of the object file, version ID of the object, whether it is the latest version, whether a delete marker is included, size of the object, last modified date of the object, ETag, storage class, cross-region replication marker, and whether the object is uploaded in multiple parts.

### Configuring the Storage Information for the Inventory Report

You need to inform COS of an export frequency of the inventory report, a bucket used to store the inventory report, and whether the inventory report needs to be encrypted. Please see the configuration details as below:

- **Select an export frequency**: Daily or weekly. COS will execute the inventory feature at the specified frequency.
- **Select an encryption mode**: No encryption or SSE-COS. If SSE-COS is selected, COS encrypts the generated inventory report.
- **Configure an export location**: You need to specify the bucket to store the inventory report.

The destination bucket and the source bucket need to be located in the same region and may be a same bucket.

## Usage

### Configuring the Inventory via Console

To learn how to configure the inventory via Console, see [Enabling Inventory](#) in the console documentation.
Configuring the Inventory via APIs

To learn how to configure the inventory via APIs, see the following API Documentation:

- PUT Bucket inventory
- GET Bucket inventory
- DELETE Bucket inventory
- List Bucket Inventory Configurations

Storage Path for the Inventory Report

The inventory report and related manifest files are published in the destination bucket. The inventory report is published in the following path:

```
destination-prefix/source-bucket/config-ID/data/
```

The related manifest files are published in the following file of the destination bucket:

```
destination-prefix/source-bucket/config-ID/YYYY-MM-DD-HH-MM/manifest.json
destination-prefix/source-bucket/config-ID/YYYY-MM-DD-HH-MM/manifest.checksum
```

Meaning of the path is as follows:

- **destination-prefix**: "Destination Prefix" set by the user during configuration of the inventory. It may be used to group all inventory reports in a public location in the destination bucket.
- **source-bucket**: Name of the source bucket corresponding to the inventory report. This folder is intended to prevent possible conflicts when inventory reports from multiple source buckets are sent to a same destination bucket.
- **config-ID**: "Inventory Name" set by the user during configuration of the inventory. When multiple inventory reports are set for a same source bucket and sent to a same destination bucket, you can specify config-ID to distinguish between the different inventory reports.
- **YYYY-MM-DD-HH-MM**: Timestamp, including the time and the date when the bucket starts to be scanned during generation of the inventory report.
- **manifest.json**: Manifest file.
- **manifest.checksum**: MD5 of the content of the manifest.json file.

The related manifest files include two files: manifest.json and manifest.checksum.

The following describes the related manifest files:

- Both manifest.json and manifest.checksum are manifest files. manifest.json describes the location of the inventory report, and manifest.checksum is the MD5 of manifest.json. Each newly delivered inventory report comes with a set of new manifest files.
- Each manifest included in manifest.json provides metadata and other basic information related to the inventory. The information includes:
  - Name of the source bucket
  - Name of the destination bucket
  - Inventory version
  - Timestamp, including the date and the time when the bucket starts to be scanned during generation of the inventory report.
  - Format and architecture of the inventory file
  - Object key, size, and MD5 checksum of the inventory report in the destination bucket.
The following shows an example of the manifest content in the manifest.json file in .csv format:

```
{
    "sourceBucket": "example-source-bucket-1250000000",
    "destinationBucket": "example-inventory-destination-bucket-1250000000",
    "fileFormat": "CSV",
    "fileSchema": "Bucket, Key, Size, StorageClass, ETag, ReplicationStatus, MultipartUploaded, LastModifiedDate, VersionId, IsLatest, IsDeleteMarker",
    "files": [
        {
            "key": "destination-prefix/example-source-bucket-1250000000/config-ID/data/939c6d46-85a9-4ba8-87bd-9db705a579ce.csv.gz",
            "size": 2147483647,
            "MD5checksum": "f11166069f1998a9eb9c97ace9cdabc"
        }
    ]
}
```

**Inventory Consistency**

All of your objects might not appear in each inventory list. The inventory list provides eventual consistency for PUTs of both new objects and overwrites, and DELETEs. Therefore, the inventory report possibly does not include the latest added or deleted object. For example, if the user uploads or deletes an object when COS is executing an inventory task configured by the user, the results of the upload or deletion operations may be not reflected in the inventory report.

To validate the status of the object before COS performs an operation, you can search for the metadata of the object by using the HEAD Object API, or check the attributes of the object in COS Console.
Bucket Tag Overview

Overview

A bucket tag is a key-value pair (key = value), which is comprised of a tag key and a tag value that are connected by an equal sign (=), for example: group = IT. It can be used to identify a bucket, helping you achieve group management of buckets. You can set, query, and delete the tag for a specified bucket.

Specifications and Limits

Limits on Tag Keys

- A tag key starting with "qcs:" or "project" is a default tag key and cannot be changed.
- A tag key can only contain characters encoded in UTF-8, spaces, numbers, or special characters including +, -, =, ., _, :, /, @.
- A tag key should be a combination of 0-127 characters encoded in UTF-8.
- Tag keys are case sensitive.

Limits on Tag Values

- A tag value can only contain characters encoded in UTF-8, spaces, numbers, or special characters including +, -, =, ., _,
- A tag value should be a combination of 0-255 characters encoded in UTF-8.
- Tag values are case sensitive.

Limits on the Number of Tags

- Bucket dimension: A resource has at most 10 different bucket tags.
- Tag dimension:
  - A user has at most 1,000 different keys.
  - A key has at most 1,000 values.
  - Multiple same keys are not allowed in the same bucket.

Usage

You can set bucket tags via the console or APIs.

Via the COS Console

For more information on how to set a bucket tag using the COS Console, see Setting Bucket Tags in Console Guide.

Via REST APIs

You can directly manage bucket tags using the following APIs:

- PUT Bucket tagging
- GET Bucket tagging
- DELETE Bucket tagging
Overview of object Tag

Overview

Object Tagging is designed to help you group and manage objects in your bucket by adding a key-value pair as object identifier. An object tag consists of a tagKey, a =, and a tagValue, such as group = IT. You can set, query and delete tags on the specified object.

Directions

Using COS Console

You can manage object tags in the COS Console. For more information, see [Setting Object Tags](https://intl.cloud.tencent.com/document/product/436/35664).

Using RESTful APIs

You can manage object tags via the following APIs:

- PUT Object tagging
- GET Object tagging
- DELETE Object tagging

Specifications and Restrictions

**Tag key restrictions**

- You cannot create tag keys starting with qcs:, project, or any other system reserved characters.
- A tag key can only contain UTF-8 encoded letters, spaces, numbers, and special characters +, -, =, ., _, :, /, @.
- A tag key can be 1-127 characters encoded in UTF-8.
- Tag keys are case-sensitive.

**Tag value restrictions**

- A tag value can only contain UTF-8-encoded letters, spaces, numbers, and special characters +, -, =, ., _, :, /, @.
- A tag value can be 1-127 characters encoded in UTF-8.
- Tag values are case-sensitive.

**Restrictions on the number of tags**

- Object dimension: up to 10 different tags for one object.
- Tag dimension: none.
Event Notifications

Last updated: 2019-09-24 17:30:12

Overview

If any change is made to COS resources (such as new files uploaded or files deleted), you will receive prompt notification. Event notification can be used together with Serverless Cloud Function to meet the needs of more application scenarios:

- **Interaction with other Tencent Cloud services**: For example, purge CDN cache or update the database when a new file is uploaded to COS.
- **System integration**: Your own service APIs can be called when files in COS are created, deleted, or overwritten. In user-generated content (UGC) scenarios, with the event notification feature, the client side will be able to interact with the server side.
- **Data processing**: Files in COS can be automatically processed, such as automatic decompression and AI recognition.

COS event notification has the following features:

- Async processing: sending notifications does not affect normal COS operations.
- Notification targets: notifications can only be sent to SCF functions in the same region.

Currently, the feature covers the following COS events:

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cos: ObjectCreated:*</td>
<td>All upload events mentioned below can trigger the SCF function</td>
</tr>
<tr>
<td>cos: ObjectCreated:Put</td>
<td>Creating a file via the Put Object API</td>
</tr>
<tr>
<td>cos: ObjectCreated:Post</td>
<td>Creating a file via the Post Object API</td>
</tr>
<tr>
<td>cos: ObjectCreated:Copy</td>
<td>Creating a file via the Put Object - Copy API</td>
</tr>
<tr>
<td>cos: ObjectCreated:CompleteMultipartUpload</td>
<td>Creating a file with the CompleteMultipartUploadt API</td>
</tr>
<tr>
<td>cos: ObjectRemove:*</td>
<td>All deletion events mentioned below can trigger the SCF function</td>
</tr>
<tr>
<td>cos: ObjectRemove:Delete</td>
<td>Deleting an object via the Delete Object API or deleting an object with a specified version using <code>versionid</code> from a bucket for which versioning is not</td>
</tr>
</tbody>
</table>
Cloud Object Storage

| cos: ObjectRemove:DeleteMarkerCreated | Deleting an object via the Delete Object API from a bucket for which versioning is enabled or suspended |

How to Use COS Event Notification

Please follow the steps below:

1. Create an SCF function
   - You can create a function in the SCF Console or using the CLI. When creating the function, you need to select the runtime environment based on the language you will use to write the function and submit the function code either by editing online or uploading a local code package.
   - You can also use a preconfigured SCF template to simplify the creation. For more information, see Create a Function. The way to write the function varies by programming language. For more information, see the SCF documentation.

2. Test the function
   Once a function is created, you can test it with a test template which can simulate COS events and trigger the function. For more information, see Testing a Function.

3. Add a trigger
   After you finish the testing, you can bind the SCF function with a bucket by creating a COS trigger in the console or using the command line. For more information, see Creating a Trigger.

4. Check if the function works
   After completing the steps above, you can make changes to the bucket in COS to see if everything works. For example, you can upload or delete files in the console or using the COS Browser and then go to SCF Console > Function Details > Execution Logs to check if everything works properly.

For more information on SCF COS triggers, see COS Trigger.
Data Extraction

SELECT Overview

The COS Select feature uses structured query language (SQL) to filter objects stored in COS in order to extract a specific object and get the desired data. By filtering object data using this feature, you can reduce the amount of data transferred by COS, which helps lower the cost and delay in data extraction.

The COS Select feature currently allows you to extract objects stored in CSV and JSON formats and compressed by gzip and bzip2 (for CSV and JSON objects only). In addition, you can save the extraction result in CSV and JSON formats and specify how to separate the result records.

You can pass in an SQL expression to COS in your request. COS Select currently only supports certain SQL expressions. For more information, see SQL Functions.

You can use the COS SDK, API, COSCMD, or COS Console to execute on SQL queries. Note that certain restrictions apply on file extraction if you use the COS Console: Up to 128 MB of files can be extracted, and up to 40 MB of data can be returned. To extract more data, use other methods.

- For more information on data types supported by COS Select and current reserved fields, see Data Types and Reserved Fields.
- Currently, the extraction function only supports public cloud regions in mainland China.

Use Limits

The following restrictions apply to COS Select:

- You must have the cos:GetObject permission to the queried object. A root account has this permission by default.
- Only objects in standard storage class can be extracted.
- The maximum length of an SQL expression is 256 KB.
- The maximum length of a single record in the extraction result is 1 MB.

SQL clauses currently supported by COS Select include:

- SELECT statement
- FROM clause
- WHERE clause
- LIMIT clause

For more information on SQL clauses, see SELECT Command.

Functions currently supported by COS Select include:

- Aggregate functions, such as AVG function, COUNT function, MAX function, MIN function, and SUM function.
- Condition functions, such as COALESCE function and NULLIF function.
- Conversion functions, such as CAST function.
- Date functions, such as DATE_ADD function, DATE_DIFF function, EXTRACT function, TO_STRING function, TO_TIMESTAMP function, and UTCNOW function.
String functions, such as CHAR_LENGTH function, CHARACTER_LENGTH function, LOWER function, SUBSTRING function, TRIM function, and UPPER function.

For more information on SQL functions, see SQL Functions.

COS Select currently supports the following operators:

- Logical operators: AND, NOT, OR
- Comparison operators: <, >, <>, >=, =, !=, BETWEEN, IN
- Pattern matching operators: LIKE
- Mathematical operators: +, -, *, %

For more information on operators, see Operators.

Initiating an extraction request

You can initiate an extraction request using the console, API, or SDK:

- If you use the console, see Data Extraction.
- If you use the SDK, go to SDK Overview and select the required SDK API.
- If you use the API, see SELECT Object Content.

FAQ

If a problem occurs when you execute on a query, COS Select will return an error code and associated error message. For the list of error codes and descriptions, see Special Error Codes.
SELECT Command

Overview

The COS Select feature only supports the SELECT command to extract the required data and reduce the amount of data transferred, which helps lower the costs and request delay. The following are the standard clauses supported by SELECT queries:

- SELECT statement
- WHERE clause
- LIMIT clause

COS Select currently does not support clause queries or joins.

SELECT Statement

The SELECT statement can extract the data you want to see from a COS object. You can query the data at different dimensions such as column name, function, and expression, and the query result will be returned as a list. The format of SELECT statement call is as follows:

```
SELECT *
SELECT projection [ AS column_alias | column_alias ] [, ...]
```

The first SELECT statement is marked with * (asterisk) and will return all the columns in the COS object. The second one uses a user-defined output scalar expression, and `projection` creates a list of outputs with custom names for each column.

WHERE Clause

The WHERE clause uses the following syntax:

```
WHERE condition
```

The WHERE clause filters data by `condition`. `condition` is an expression that returns a Boolean result, and only rows with a return value of TRUE will be output in the result.

LIMIT Clause

The LIMIT clause uses the following syntax:

```
LIMIT number
```

The LIMIT clause sets a limit on the number of records to be returned per query, which can be specified using the `number` parameter.
The SELECT and WHERE clauses can select the fields to be queried in any of the following ways, depending on whether the file format is CSV or JSON.

**CSV**

- **Column number**: You can use \_N to specify the data in column N for query. For any CSV files, the column number increases from 1. For example, the first column is numbered \_1, and the second column is numbered \_2. In the SELECT and WHERE clauses, it is valid to specify the column to be queried using \_N or alias\_N.

- **Column header**: If the CSV file to be queried contains column headers, the SELECT and WHERE clauses can use the headers to specify the columns to be queried, which can be specified using alias.column_name or column_name in the SELECT and WHERE clauses in an SQL statement.

**JSON**

- **Document**: You can access a JSON document using alias.name. A nested array can be accessed in a way such as alias.name1.name2.name3.

- **List**: You can access the elements in a list using an index, which is numbered from 0 and uses the [] operator. For example, you can access the second element in a JSON list using alias[1]. If you need to access a nested array, you can also do so in a way such as alias.name1.name2[1].name3.

**Samples**

The following is the sample data in the samples:

```json
{
    "name": "Leon",
    "org": "Tencent",
    "projects": [
        {"project_name": "project1", "completed": true},
        {"project_name": "project2", "completed": false}
    ]
}
```

- **Sample 1.** The following is the SQL statement used to query \_name\_ in the sample data and the query result:

  ```sql
  Select s.name from COSObject s
  
  ("name": "Leon")
  ```

- **Sample 2.** The following is the SQL statement used to query \_project\_name\_ in the sample data and the query result:

  ```sql
  Select s.projects[0].project_name from COSObject s
  
  ("project_name": "project1")
  ```

**Case Sensitivity of Headers and Attribute Names**

You can use double quotation marks to indicate whether headers in a CSV file and attribute names in a JSON file are case-sensitive. If no double quotation marks are added, the headers/attribute names are case-insensitive. If this is not explicitly specified, COS Select may throw an exception.

- **Sample 1.** Query objects with a header/attribute name containing "NAME".

  Because the following sample SQL query does not contain double quotation marks, the query is case-insensitive. As this header is present in the table, a value will be successfully returned eventually.
SELECT s.name from COSObject s

Because the following sample SQL query contains double quotation marks, the query is case-sensitive. As the table does not contain this header, the `SQLParsingError` 400 error will be eventually returned.

SELECT s."name" from COSObject s

Sample 2. Query objects with a header/attribute name containing "NAME" and "name".
Because the following sample SQL query does not contain double quotation marks, the query is case-insensitive. As the table contains two headers "NAME" and "name", the query command is ambiguous, and the `AmbiguousFieldName` exception will be thrown.

SELECT s.name from COSObject s

Because the following sample SQL query contains double quotation marks, the query is case-sensitive. As the table contains the header "NAME", the query result will be successfully returned.

SELECT s."NAME" from COSObject s

Using Reserved Fields as User-defined Fields

The SQL expressions of COS Select have certain reserved fields such as function name, data type, and operator. Sometimes you probably use these reserved fields as column headers in a CSV file or attribute names in a JSON, which may cause conflicts with reserved fields. In this case, you can use double quotation marks to indicate that you are using a custom field; otherwise, COS will return 400 parse error.

For the complete list of reserved fields, see [Reserved Words](#).

- Sample: The header/attribute name in the object to be queried contains a reserved field "CAST".
  The following sample SQL query uses double quotation marks to indicate that CAST is a user-defined field, so the query result will be successfully returned.

SELECT s."CAST" from COSObject s

The following sample SQL query does not use double quotation marks to indicate that CAST is a user-defined field, so COS will treat it as a reserved field and return 400 parse error.

SELECT s.CAST from COSObject s

Scalar Expressions

In the SELECT statement and the WHERE clause, you can use SQL scalar expressions (expressions that return a scalar). Currently, COS Select supports the following forms:

- **literal**: SQL text.
- **column_reference**: column_name or alias.column_name.
- **unary_op expression**: SQL unary operator.
- **expression binary_op expression**: SQL binary operator.
- **func_name**: Name of the called scalar function.
- `expression [ NOT ] BETWEEN expression AND expression`
- `expression LIKE expression [ ESCAPE expression ]`
SQL Functions

Last updated: 2019-10-25 16:11:39

Aggregate Functions

COS Select supports the following aggregate functions:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Parameter Type</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVG(expression)</td>
<td>INT, FLOAT, and DECIMAL</td>
<td>DECIMAL will be returned if the input parameter is of integer type, and FLOAT if float type. The same type as the input parameter will be returned in all other cases.</td>
</tr>
<tr>
<td>COUNT</td>
<td>-</td>
<td>INT</td>
</tr>
<tr>
<td>MAX(expression)</td>
<td>INT and DECIMAL</td>
<td>The return value type is the same as that of the input parameter</td>
</tr>
<tr>
<td>MIN(expression)</td>
<td>INT and DECIMAL</td>
<td>The return value type is the same as that of the input parameter</td>
</tr>
<tr>
<td>SUM(expression)</td>
<td>INT, FLOAT, DOUBLE, and DECIMAL</td>
<td>INT will be returned if the input parameter is of integer type, and FLOAT if float type. The same type as the input parameter will be returned in all other cases.</td>
</tr>
</tbody>
</table>

Condition Functions

COS Select supports the following condition functions:

**COALESCE**

The COALESCE function determines the input parameters in sequence and returns the first non-null parameter value. If the input parameters do not include a non-null parameter, the function will return a null value.

**Syntax**

COALESCE ( expression, expression, ... )

Values, arrays, or nested functions of INT, String, and Float types can be passed in for the expression parameter.

**Samples**

<table>
<thead>
<tr>
<th>Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>COALESCE(1) -- 1</td>
</tr>
<tr>
<td>COALESCE(1, null) -- 1</td>
</tr>
<tr>
<td>COALESCE(null, null, 1) -- 1</td>
</tr>
<tr>
<td>COALESCE(missing, 1) -- 1</td>
</tr>
<tr>
<td>COALESCE(null, 'string') -- 'string'</td>
</tr>
<tr>
<td>COALESCE(null) -- null</td>
</tr>
<tr>
<td>COALESCE(null, null) -- null</td>
</tr>
<tr>
<td>COALESCE(missing) -- null</td>
</tr>
<tr>
<td>COALESCE(missing, missing) -- null</td>
</tr>
</tbody>
</table>

**NULLIF**

The NULLIF function determines the difference between two parameters passed in. If the two parameters have the same value, NULL will be returned; otherwise, the value of the first parameter passed in will be returned.
Syntax

```
NULLIF ( expression1, expression2 )
```

Values, arrays, or nested functions of INT, String, and Float types can be passed in for the `expression` parameter.

Samples

```
NULLIF(1, 2) -- 1
NULLIF(1, '1') -- 1
NULLIF(1, NULL) -- 1
NULLIF(1, 1) -- null
NULLIF(1.0, 1) -- null
NULLIF(missing, null) -- null
NULLIF(missing, missing) -- null
NULLIF([1], [1]) -- null
NULLIF(NULL, 1) -- null
NULLIF(null, null) -- null
```

Conversion Functions

COS Select supports the following conversion functions:

**CAST**

The CAST function converts one instance to another instance. The instance can be either a value or a function that can be calculated to a certain value.

Syntax

```
CAST ( expression AS data_type )
```

- The `expression` parameter can be a value, an array, an operator, or an SQL function that can be calculated to a certain value.
- The `data_type` parameter is the data type after conversion, such as INT. For the data types currently supported by COS Select, see Data Types.

Samples

```
CAST('2007-04-05T14:30Z' AS TIMESTAMP)
CAST(0.456 AS FLOAT)
```

Date Functions

COS Select supports the following date functions:

**DATE_ADD**

The DATE_ADD function adds a specified time interval to a part (year, month, day, hour, minute, or second) of a specific timestamp and returns a new timestamp.
Syntax

```sql
DATE_ADD( date_part, quantity, timestamp )
```

- The `date_part` parameter specifies the part of the timestamp to be modified, which can be year, month, day, hour, minute, or second.
- The `quantity` parameter represents the value to be added, which must be a positive integer.
- The `timestamp` parameter represents the timestamp to be modified.

Samples

```sql
DATE_ADD(year, 5, '2010-01-01T') -- 2015-01-01
DATE_ADD(month, 1, '2010T') -- 2010-02T
DATE_ADD(month, 13, '2010T') -- 2011-02T
DATE_ADD(day, -1, '2017-01-01T') -- 2017-01-09
DATE_ADD(hour, 1, '2017T') -- 2017-01-01T01:00:00:00
DATE_ADD(hour, 1, '2017-01-02T03:04Z') -- 2017-01-02T04:04Z
DATE_ADD(minute, 1, '2017-01-02T03:04:05.006Z') -- 2017-01-02T03:05:05.006Z
DATE_ADD(second, 1, '2017-01-02T03:04:05.006Z') -- 2017-01-02T03:04:06.006Z
```

**DATE_DIFF**

The `DATE_DIFF` function compares two valid timestamps and returns the difference between them, which can be displayed in the specified unit of time. If the `date_part` value of `timestamp1` is greater than that of `timestamp2`, a positive number will be returned; otherwise, a negative number will be returned.

Syntax

```sql
DATE_DIFF( date_part, timestamp1, timestamp2 )
```

- The `date_part` parameter specifies the unit of time which the two timestamps are compared in and can be year, month, day, hour, minute, or second.
- The `timestamp1` parameter is the first input timestamp.
- The `timestamp2` parameter is the second input timestamp.

Samples

```sql
DATE_DIFF(year, '2010-01-01T', '2011-01-01T') -- 1
DATE_DIFF(year, '2010T', '2010-05T') -- 4
DATE_DIFF(month, '2010T', '2011T') -- 12
DATE_DIFF(month, '2011T', '2010T') -- -12
DATE_DIFF(day, '2010-01-01T23:00T', '2010-01-02T01:00T') -- 0
```

**EXTRACT**

The `EXTRACT` function extracts a value in the specified unit of time from a given timestamp.

Syntax

```sql
EXTRACT( date_part FROM timestamp )
```
The parameter `date_part` specifies the unit of time to be extracted, which can be year, month, day, hour, minute, or second. The `timestamp` parameter represents the input timestamp.

**Samples**

```sql
EXTRACT(YEAR FROM '2010-01-01T') -- 2010
EXTRACT(MONTH FROM '2010T') -- 1
EXTRACT(MONTH FROM '2010-10T') -- 10
EXTRACT(HOUR FROM '2017-01-02T03:04:05+07:08') -- 3
EXTRACT(MINUTE FROM '2017-01-02T03:04:05+07:08') -- 4
EXTRACT(TIMEZONE_HOUR FROM '2017-01-02T03:04:05+07:08') -- 7
EXTRACT(TIMEZONE_MINUTE FROM '2017-01-02T03:04:05+07:08') -- 8
```

**TO_STRING**

The TO_STRING function converts a timestamp to a string of time in the specified format.

**Syntax**

```sql
TO_STRING ( timestamp time_format_pattern )
```

- The `timestamp` parameter specifies the timestamp to be converted.
- The `time_format_pattern` parameter specifies the time format.

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
<th>Sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>yy</td>
<td>2-digit year</td>
<td>98</td>
</tr>
<tr>
<td>y</td>
<td>4-digit year</td>
<td>1998</td>
</tr>
<tr>
<td>yyyy</td>
<td>Year expressed by 4 digits. If there are less than 4 digits, 0 will be automatically added</td>
<td>0199</td>
</tr>
<tr>
<td>M</td>
<td>Month</td>
<td>1</td>
</tr>
<tr>
<td>MM</td>
<td>Month expressed by 2 digits. If there are less than 2 digits, 0 will be automatically added</td>
<td>01</td>
</tr>
<tr>
<td>MMM</td>
<td>English abbreviation of a month</td>
<td>Jan</td>
</tr>
<tr>
<td>MMMM</td>
<td>Full English name of a month</td>
<td>January</td>
</tr>
<tr>
<td>MMMMM</td>
<td>Initial of a month</td>
<td>J (not applicable to the to_timestamp function)</td>
</tr>
<tr>
<td>d</td>
<td>Day (1-31) in a month</td>
<td>1</td>
</tr>
<tr>
<td>dd</td>
<td>Day expressed by 2 digits (1-31)</td>
<td>01</td>
</tr>
<tr>
<td>a</td>
<td>Symbol for morning or afternoon (AM/PM)</td>
<td>AM</td>
</tr>
<tr>
<td>h</td>
<td>Hour in 12-hour time</td>
<td>1</td>
</tr>
<tr>
<td>hh</td>
<td>Hour expressed by 2 digits in 12-hour time</td>
<td>01</td>
</tr>
<tr>
<td>H</td>
<td>Hour in 24-hour time</td>
<td>1</td>
</tr>
<tr>
<td>Format</td>
<td>Description</td>
<td>Sample</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>HH</td>
<td>Hour expressed by 2 digits in 24-hour time</td>
<td>01</td>
</tr>
<tr>
<td>m</td>
<td>Minute (00-59)</td>
<td>1</td>
</tr>
<tr>
<td>mm</td>
<td>Minute expressed by 2 digits in 24-hour time</td>
<td>01</td>
</tr>
<tr>
<td>s</td>
<td>Second (00-59)</td>
<td>1</td>
</tr>
<tr>
<td>ss</td>
<td>Second expressed by 2 digits in 24-hour time</td>
<td>01</td>
</tr>
<tr>
<td>S</td>
<td>Decimal part of the second (accuracy: 0.1; value range: 0.0 - 0.9)</td>
<td>0</td>
</tr>
<tr>
<td>SS</td>
<td>Decimal part of the second (accuracy: 0.01; value range: 0.00 - 0.99)</td>
<td>6</td>
</tr>
<tr>
<td>SSS</td>
<td>Decimal part of the second (accuracy: 0.001; value range: 0.000 - 0.999)</td>
<td>60</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>SSSSSSSS</td>
<td>Decimal part of the second (accuracy: 0.000000001; value range: 0.000000000 - 0.999999999)</td>
<td>60000000</td>
</tr>
<tr>
<td>n</td>
<td>Nanosecond</td>
<td>60000000</td>
</tr>
<tr>
<td>X</td>
<td>Hour-level offset. If the offset is 0, then this will be &quot;Z&quot;</td>
<td>+01 or Z</td>
</tr>
<tr>
<td>XX or XXX</td>
<td>Hour- or minute-level offset. If the offset is 0, then this will be &quot;Z&quot;</td>
<td>+0100 or Z</td>
</tr>
<tr>
<td>xxx or xxxxx</td>
<td>Hour- or minute-level offset. If the offset is 0, then this will be &quot;Z&quot;</td>
<td>+01:00 or Z</td>
</tr>
<tr>
<td>x</td>
<td>Hour-level offset</td>
<td>1</td>
</tr>
<tr>
<td>xx or xxx</td>
<td>Hour- or minute-level offset</td>
<td>0100</td>
</tr>
<tr>
<td>xxx or xxxxx</td>
<td>Hour- or minute-level offset</td>
<td>01:00</td>
</tr>
</tbody>
</table>

**Samples**

```
TO_STRING(’1998-07-20T20:18Z’, ’M-d-yy’) -- "7-20-69"
TO_STRING(’1998-07-20T20:18Z’, ’MMM d, y:m a’) -- "July 20, 1998 8:18 PM"
TO_STRING(’1998-07-20T20:18+08:00Z’, ’y-MM-dd’T’H:mm:ssX’) -- "1998-07-20T20:18:00+0800"
TO_STRING(’1998-07-20T20:18+08:00’, ’y-MM-dd’T’H:mm:ssXXX’) -- "1998-07-20T20:18:00+08:00"
```

**TO_TIMESTAMP**

The TO_TIMESTAMP function converts a string of time to a timestamp.

**Syntax**

```
TO_TIMESTAMP ( string )
```

The `string` parameter represents the input time string.
Samples

```sql
TO_TIMESTAMP('2007T') -- '2007T'
TO_TIMESTAMP('2007-02-23T12:14:33.079-08:00') -- '2007-02-23T12:14:33.079-08:00'
```

**UTCNOW**

The UTCNOW function returns the current timestamp in UTC.

**Syntax**

```sql
UTCNOW()
```

**Samples**

```sql
```

**String Functions**

COS Select supports the following string functions:

**CHAR_LENGTH, CHARACTER_LENGTH**

Both CHAR_LENGTH and CHARACTER_LENGTH can compute the number of characters in a string, and they have the same semantics.

**Syntax**

```sql
CHAR_LENGTH ( string )
```

The `string` parameter specifies the string for character counting.

**Samples**

```sql
CHAR_LENGTH('null') -- 4
CHAR_LENGTH('tencent') -- 7
```

**LOWER**

The LOWER function converts all uppercase letters in the specified string to lowercase letters, with all non-uppercase letters left unchanged.

**Syntax**

```sql
LOWER ( string )
```

The `string` parameter specifies the string for which to convert uppercase letters to lowercase letters.

**Samples**
LOWER('TENcent') -- 'tencent'

**SUBSTRING**

The SUBSTRING function returns a substring of a string. You can specify an index from which the SUBSTRING function will extract the remainder of the original string based on the length of the specified substring and return the result.

If the input string contains only 1 character, and the index is set to greater than 1, the SUBSTRING function will automatically switch it to 1.

**Syntax**

```
SUBSTRING( string FROM start [ FOR length ] )
```

- The `string` parameter specifies the string from which to extract a substring.
- The `start` parameter represents an index value of the string as the starting position for extraction.
- The `length` parameter specifies the length of the substring. If the length of the substring is not specified, the remainder of the string will be extracted.

**Samples**

```
SUBSTRING("123456789", 0) -- "123456789"
SUBSTRING("123456789", 1) -- "123456789"
SUBSTRING("123456789", 2) -- "23456789"
SUBSTRING("123456789", -4) -- "123456789"
SUBSTRING("123456789", 0, 999) -- "123456789"
SUBSTRING("123456789", 1, 5) -- "12345"
```

**TRIM**

The TRIM function deletes all characters before the first character or after the last character of the specified string. " " is the default character to be deleted.

**Syntax**

```
TRIM ( [ [LEADING | TRAILING | BOTH remove_chars] FROM] string )
```

- The `string` parameter specifies the string to be manipulated.
- The `LEADING | TRAILING | BOTH` parameter specifies the extra characters to be deleted, which can be before the string (LEADING), after the string (TRAILING), or both (BOTH).
- The `remove_chars` parameter specifies the type of extra characters to be deleted. It can be a string containing more than one characters. The TRIM function will delete all extra characters of the corresponding type that are identified by the TRIM function before or after the `string` parameter.

**Samples**

```
TRIM(' foobar ') -- 'foobar'
TRIM(' $foobar$' ) -- '$foobar$'
TRIM(LEADING FROM ' foobar ' ) -- 'foobar'
```
TRIM(TRAILING FROM 'foobar') -- 'foobar'
TRIM(BOTH FROM 'foobar') -- 'foobar'
TRIM(BOTH '12' FROM '1112211foobar22211122') -- 'foobar'

**UPPER**

The UPPER function converts all lowercase letters to uppercase letters with non-lowercase letters left unchanged.

**Syntax**

```
UPPER ( string )
```

The `string` parameter specifies the string to be converted to uppercase letters.

**Samples**

```
UPPER('tenCENT') -- 'TENCENT'
```
Reserved Words

In order to ensure normal operation and subsequent expansion, the COS Select feature has the following reserved fields such as function name, data type, and operator. You may use these fields to facilitate your SQL queries.

<table>
<thead>
<tr>
<th>No.</th>
<th>Field</th>
<th>No.</th>
<th>Field</th>
<th>No.</th>
<th>Field</th>
<th>No.</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>absolute</td>
<td>51</td>
<td>create</td>
<td>101</td>
<td>goto</td>
<td>151</td>
<td>octet_length</td>
</tr>
<tr>
<td>2</td>
<td>action</td>
<td>52</td>
<td>cross</td>
<td>102</td>
<td>grant</td>
<td>152</td>
<td>of</td>
</tr>
<tr>
<td>3</td>
<td>add</td>
<td>53</td>
<td>current</td>
<td>103</td>
<td>group</td>
<td>153</td>
<td>on</td>
</tr>
<tr>
<td>4</td>
<td>all</td>
<td>54</td>
<td>current_date</td>
<td>104</td>
<td>having</td>
<td>154</td>
<td>only</td>
</tr>
<tr>
<td>5</td>
<td>allocate</td>
<td>55</td>
<td>current_time</td>
<td>105</td>
<td>hour</td>
<td>155</td>
<td>open</td>
</tr>
<tr>
<td>6</td>
<td>alter</td>
<td>56</td>
<td>current_timestamp</td>
<td>106</td>
<td>identity</td>
<td>156</td>
<td>option</td>
</tr>
<tr>
<td>7</td>
<td>and</td>
<td>57</td>
<td>current_user</td>
<td>107</td>
<td>immediate</td>
<td>157</td>
<td>or</td>
</tr>
<tr>
<td>8</td>
<td>any</td>
<td>58</td>
<td>cursor</td>
<td>108</td>
<td>in</td>
<td>158</td>
<td>order</td>
</tr>
<tr>
<td>9</td>
<td>are</td>
<td>59</td>
<td>date</td>
<td>109</td>
<td>indicator</td>
<td>159</td>
<td>outer</td>
</tr>
<tr>
<td>10</td>
<td>as</td>
<td>60</td>
<td>day</td>
<td>110</td>
<td>initially</td>
<td>160</td>
<td>output</td>
</tr>
<tr>
<td>11</td>
<td>asc</td>
<td>61</td>
<td>deallocate</td>
<td>111</td>
<td>inner</td>
<td>161</td>
<td>overlaps</td>
</tr>
<tr>
<td>12</td>
<td>assertion</td>
<td>62</td>
<td>dec</td>
<td>112</td>
<td>input</td>
<td>162</td>
<td>pad</td>
</tr>
<tr>
<td>13</td>
<td>at</td>
<td>63</td>
<td>decimal</td>
<td>113</td>
<td>insensitive</td>
<td>163</td>
<td>partial</td>
</tr>
<tr>
<td>14</td>
<td>authorization</td>
<td>64</td>
<td>declare</td>
<td>114</td>
<td>insert</td>
<td>164</td>
<td>pivot</td>
</tr>
<tr>
<td>15</td>
<td>avg</td>
<td>65</td>
<td>default</td>
<td>115</td>
<td>int</td>
<td>165</td>
<td>position</td>
</tr>
<tr>
<td>16</td>
<td>bag</td>
<td>66</td>
<td>deferrable</td>
<td>116</td>
<td>integer</td>
<td>166</td>
<td>precision</td>
</tr>
<tr>
<td>17</td>
<td>begin</td>
<td>67</td>
<td>deferred</td>
<td>117</td>
<td>intersect</td>
<td>167</td>
<td>prepare</td>
</tr>
<tr>
<td>18</td>
<td>between</td>
<td>68</td>
<td>delete</td>
<td>118</td>
<td>interval</td>
<td>168</td>
<td>preserve</td>
</tr>
<tr>
<td>19</td>
<td>bit</td>
<td>69</td>
<td>desc</td>
<td>119</td>
<td>into</td>
<td>169</td>
<td>primary</td>
</tr>
<tr>
<td>20</td>
<td>bit_length</td>
<td>70</td>
<td>describe</td>
<td>120</td>
<td>is</td>
<td>170</td>
<td>prior</td>
</tr>
<tr>
<td>21</td>
<td>blob</td>
<td>71</td>
<td>descriptor</td>
<td>121</td>
<td>isolation</td>
<td>171</td>
<td>privileges</td>
</tr>
<tr>
<td>22</td>
<td>bool</td>
<td>72</td>
<td>diagnostics</td>
<td>122</td>
<td>join</td>
<td>172</td>
<td>procedure</td>
</tr>
<tr>
<td>23</td>
<td>boolean</td>
<td>73</td>
<td>disconnect</td>
<td>123</td>
<td>key</td>
<td>173</td>
<td>public</td>
</tr>
<tr>
<td>24</td>
<td>both</td>
<td>74</td>
<td>distinct</td>
<td>124</td>
<td>language</td>
<td>174</td>
<td>read</td>
</tr>
<tr>
<td>25</td>
<td>by</td>
<td>75</td>
<td>domain</td>
<td>125</td>
<td>last</td>
<td>175</td>
<td>real</td>
</tr>
<tr>
<td>26</td>
<td>cascade</td>
<td>76</td>
<td>double</td>
<td>126</td>
<td>leading</td>
<td>176</td>
<td>references</td>
</tr>
<tr>
<td>27</td>
<td>cascaded</td>
<td>77</td>
<td>drop</td>
<td>127</td>
<td>left</td>
<td>177</td>
<td>relative</td>
</tr>
<tr>
<td>28</td>
<td>case</td>
<td>78</td>
<td>else</td>
<td>128</td>
<td>level</td>
<td>178</td>
<td>restrict</td>
</tr>
</tbody>
</table>

Reserved Words Last updated: 2019-10-25 16:12:26
<table>
<thead>
<tr>
<th>No.</th>
<th>Field</th>
<th>No.</th>
<th>Field</th>
<th>No.</th>
<th>Field</th>
<th>No.</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>cast</td>
<td>79</td>
<td>end</td>
<td>129</td>
<td>like</td>
<td>179</td>
<td>revoke</td>
</tr>
<tr>
<td>30</td>
<td>catalog</td>
<td>80</td>
<td>end-exec</td>
<td>130</td>
<td>limit</td>
<td>180</td>
<td>right</td>
</tr>
<tr>
<td>31</td>
<td>char</td>
<td>81</td>
<td>escape</td>
<td>131</td>
<td>list</td>
<td>181</td>
<td>rollback</td>
</tr>
<tr>
<td>32</td>
<td>char_length</td>
<td>82</td>
<td>except</td>
<td>132</td>
<td>local</td>
<td>182</td>
<td>rows</td>
</tr>
<tr>
<td>33</td>
<td>character</td>
<td>83</td>
<td>exception</td>
<td>133</td>
<td>lower</td>
<td>183</td>
<td>schema</td>
</tr>
<tr>
<td>34</td>
<td>character_length</td>
<td>84</td>
<td>exec</td>
<td>134</td>
<td>match</td>
<td>184</td>
<td>scroll</td>
</tr>
<tr>
<td>35</td>
<td>check</td>
<td>85</td>
<td>execute</td>
<td>135</td>
<td>max</td>
<td>185</td>
<td>second</td>
</tr>
<tr>
<td>36</td>
<td>clob</td>
<td>86</td>
<td>exists</td>
<td>136</td>
<td>min</td>
<td>186</td>
<td>section</td>
</tr>
<tr>
<td>37</td>
<td>close</td>
<td>87</td>
<td>external</td>
<td>137</td>
<td>minute</td>
<td>187</td>
<td>select</td>
</tr>
<tr>
<td>38</td>
<td>coalesce</td>
<td>88</td>
<td>extract</td>
<td>138</td>
<td>missing</td>
<td>188</td>
<td>session</td>
</tr>
<tr>
<td>39</td>
<td>collate</td>
<td>89</td>
<td>false</td>
<td>139</td>
<td>module</td>
<td>189</td>
<td>session_user</td>
</tr>
<tr>
<td>40</td>
<td>collation</td>
<td>90</td>
<td>fetch</td>
<td>140</td>
<td>month</td>
<td>190</td>
<td>set</td>
</tr>
<tr>
<td>41</td>
<td>column</td>
<td>91</td>
<td>first</td>
<td>141</td>
<td>names</td>
<td>191</td>
<td>sexp</td>
</tr>
<tr>
<td>42</td>
<td>commit</td>
<td>92</td>
<td>float</td>
<td>142</td>
<td>national</td>
<td>192</td>
<td>size</td>
</tr>
<tr>
<td>43</td>
<td>connect</td>
<td>93</td>
<td>for</td>
<td>143</td>
<td>natural</td>
<td>193</td>
<td>smallint</td>
</tr>
<tr>
<td>44</td>
<td>connection</td>
<td>94</td>
<td>foreign</td>
<td>144</td>
<td>nchar</td>
<td>194</td>
<td>some</td>
</tr>
<tr>
<td>45</td>
<td>constraint</td>
<td>95</td>
<td>found</td>
<td>145</td>
<td>next</td>
<td>195</td>
<td>space</td>
</tr>
<tr>
<td>46</td>
<td>constraints</td>
<td>96</td>
<td>from</td>
<td>146</td>
<td>no</td>
<td>196</td>
<td>sql</td>
</tr>
<tr>
<td>47</td>
<td>continue</td>
<td>97</td>
<td>full</td>
<td>147</td>
<td>not</td>
<td>197</td>
<td>sqrcode</td>
</tr>
<tr>
<td>48</td>
<td>convert</td>
<td>98</td>
<td>get</td>
<td>148</td>
<td>null</td>
<td>198</td>
<td>sqlerror</td>
</tr>
<tr>
<td>49</td>
<td>corresponding</td>
<td>99</td>
<td>global</td>
<td>149</td>
<td>nullif</td>
<td>199</td>
<td>sqlstate</td>
</tr>
<tr>
<td>50</td>
<td>count</td>
<td>100</td>
<td>go</td>
<td>150</td>
<td>numeric</td>
<td>200</td>
<td>string</td>
</tr>
</tbody>
</table>
Data Types


Overview

COS Select supports a wide variety of primitive data types.

The data type directly supported by a compiler is called a **primitive data type**.

Data Type Conversion

COS Select uses the CAST function to determine the data type of your input data. In general, if you do not specify a data type through the CAST function, COS Select will treat the input data as string type.

For more information on the CAST function, see the **CAST** section in the SQL function documentation.

Supported Data Types

COS Select supports the following primitive data types:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>TRUE/FALSE</td>
<td>FALSE</td>
</tr>
<tr>
<td>int, integer</td>
<td>An 8-byte signed integer</td>
<td>100000</td>
</tr>
<tr>
<td>string</td>
<td>A UTF-8-encoded string with a character length in the range of 1 - 2,147,483,647</td>
<td>'xyz'</td>
</tr>
<tr>
<td>float</td>
<td>An 8-byte floating point</td>
<td>CAST(0.456 AS FLOAT)</td>
</tr>
<tr>
<td>decimal, numeric</td>
<td>A decimal value with a maximum precision of 38 decimal places and in the range of $-2^{31}$ - $2^{31}$ - 1</td>
<td>123.456</td>
</tr>
<tr>
<td>timestamp</td>
<td>A timestamp represents a certain moment and can be expressed with any precision. A timestamp in text format follows the W3C specification but needs to end with &quot;T&quot; (unless recorded in days). When a fractional second is used, it should be accurate to at least one decimal place (with no upper limit on the number of decimal places). The offset of local time can be expressed by the offset expressed by hours and minutes when compared to UTC, or &quot;Z&quot; can be used to express the offset of local time from UTC. The time offset does not need to be displayed when only the date is recorded.</td>
<td>CAST('2007-04-05T14:30Z' AS TIMESTAMP)</td>
</tr>
</tbody>
</table>
Operators

Last updated: 2019-10-25 16:04:52

COS Select supports the following operators:

<table>
<thead>
<tr>
<th>Operator Type</th>
<th>Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logical operator</td>
<td>AND, NOT, OR</td>
</tr>
<tr>
<td>Comparison operator</td>
<td>&lt;, &gt;, &lt;=, &gt;=, =, !=, BETWEEN, IN</td>
</tr>
<tr>
<td>Pattern matching operator</td>
<td>LIKE</td>
</tr>
<tr>
<td>Mathematical operator</td>
<td>+, -, *, %</td>
</tr>
</tbody>
</table>

Priority of Operators

The following table shows the priority of operators in descending order:

<table>
<thead>
<tr>
<th>Operator/Element</th>
<th>Association</th>
<th>Use Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>Right</td>
<td>Unary subtraction</td>
</tr>
<tr>
<td>*, /, %</td>
<td>Left</td>
<td>Multiplication, division, and modulo</td>
</tr>
<tr>
<td>+, -</td>
<td>Left</td>
<td>Addition and subtraction</td>
</tr>
<tr>
<td>IN</td>
<td>-</td>
<td>Setting membership</td>
</tr>
<tr>
<td>BETWEEN</td>
<td>-</td>
<td>In the range of ()</td>
</tr>
<tr>
<td>LIKE</td>
<td>-</td>
<td>String pattern matching</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>-</td>
<td>Less than and greater than</td>
</tr>
<tr>
<td>=</td>
<td>Right</td>
<td>Equivalence and assignment</td>
</tr>
<tr>
<td>NOT</td>
<td>Right</td>
<td>Logical NOT</td>
</tr>
<tr>
<td>AND</td>
<td>Left</td>
<td>Logical AND</td>
</tr>
<tr>
<td>OR</td>
<td>Left</td>
<td>Logical OR</td>
</tr>
</tbody>
</table>
Overview

The log management records the detailed access information of the specified source bucket and store the information as log files in the specified bucket to facilitate bucket management.

In the destination bucket, the log path is:

```{Destination bucket}/{Path prefix} {YYYY}/{MM}/{DD}/{time}_{random}_{index}.gz```

The log is generated every 5 minutes, one record is a row, each record contains multiple fields, and the fields are separated by spaces. It should be noted that a single log file can be up to 256MB. If you generate more than 256MB of logs in these 5 minutes, your log will be split into multiple log files. The currently supported log fields are as follows:

<table>
<thead>
<tr>
<th>Field Number</th>
<th>Name</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>eventVersion</td>
<td>Log version</td>
<td>1.0</td>
</tr>
<tr>
<td>2</td>
<td>bucketName</td>
<td>Bucket name</td>
<td>examplebucket-125000000</td>
</tr>
<tr>
<td>3</td>
<td>qcsRegion</td>
<td>Request region</td>
<td>ap-beijing</td>
</tr>
<tr>
<td>4</td>
<td>eventTime</td>
<td>Event time (request end time, which is a timestamp in UTC+0 time zone)</td>
<td>2018-12-01T11:02:33Z</td>
</tr>
<tr>
<td>5</td>
<td>eventSource</td>
<td>Domain name is accessed by the user</td>
<td>examplebucket-125000000.cos.ap-guangzhou.myqcloud.com</td>
</tr>
<tr>
<td>6</td>
<td>eventName</td>
<td>Event name</td>
<td>UploadPart</td>
</tr>
<tr>
<td>7</td>
<td>remoteIp</td>
<td>Source IP</td>
<td>192.168.0.1</td>
</tr>
<tr>
<td>8</td>
<td>userSecretKeyId</td>
<td>User access KeyId</td>
<td>AKIDNYVCDojQyGj5brTf</td>
</tr>
<tr>
<td>9</td>
<td>reservedFiled</td>
<td>reserved fields</td>
<td>reserved fields, shown as -</td>
</tr>
<tr>
<td>10</td>
<td>reqBytesSent</td>
<td>Request bytes</td>
<td>83886080</td>
</tr>
<tr>
<td>11</td>
<td>deltaDataSize</td>
<td>Change in storage made by the request (in bytes)</td>
<td>808</td>
</tr>
<tr>
<td>12</td>
<td>reqPath</td>
<td>Requested file path</td>
<td>/folder/text.txt</td>
</tr>
<tr>
<td>13</td>
<td>reqMethod</td>
<td>Request method</td>
<td>put</td>
</tr>
<tr>
<td>14</td>
<td>userAgent</td>
<td>User UA</td>
<td>cos-go-sdk-v5.2.9</td>
</tr>
<tr>
<td>15</td>
<td>resHttpCode</td>
<td>HTTP return code</td>
<td>404</td>
</tr>
<tr>
<td>16</td>
<td>resErrorCode</td>
<td>Error code</td>
<td>NoSuchKey</td>
</tr>
<tr>
<td>17</td>
<td>resErrorMsg</td>
<td>Error message</td>
<td>The specified key does not exist.</td>
</tr>
<tr>
<td>18</td>
<td>resBytesSent</td>
<td>Bytes returned</td>
<td>197</td>
</tr>
<tr>
<td>Field Number</td>
<td>Name</td>
<td>Meaning</td>
<td>Example</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>19</td>
<td>resTotalTime</td>
<td>Total time used by the request (in milliseconds, i.e., the time between</td>
<td>4295</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the last byte of the response and the first byte of the request)</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>logSourceType</td>
<td>Log source type</td>
<td>USER (User access request), CDN (CDN Origin-pull request)</td>
</tr>
<tr>
<td>21</td>
<td>storageClass</td>
<td>Storage class</td>
<td>STANDARD, STANDARD_IA, ARCHIVE</td>
</tr>
<tr>
<td>22</td>
<td>accountId</td>
<td>Bucket owner ID</td>
<td>100000000001</td>
</tr>
<tr>
<td>23</td>
<td>resTurnAroundTime</td>
<td>Time used by the request server (in milliseconds, i.e., the time between</td>
<td>4295</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the first byte of the response and the last byte of the request)</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>requester</td>
<td>Visitor</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>requestId</td>
<td>Request ID</td>
<td>NWQ1ZjY4MTBfMjZiMjU4NjRfOWI1N180NDBiYTY=</td>
</tr>
<tr>
<td>26</td>
<td>objectSize</td>
<td>Object size in bytes</td>
<td>808, if you use Multipart Upload, the objectSize field will only be displayed when the upload is completed. This field displays - during each Multipart Upload.</td>
</tr>
<tr>
<td>27</td>
<td>versionId</td>
<td>Object version ID</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>targetStorageClass</td>
<td>Destination storage class, recorded for replication requests</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>STANDARD, STANDARD_IA, ARCHIVE</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>referer</td>
<td>Origin server address</td>
<td>.example.com or 111.111.111.1</td>
</tr>
<tr>
<td>31</td>
<td>requestUri</td>
<td>Request URI</td>
<td>&quot;GET /fdgfdgsf%20/%E6%B5%AE%E7%82%B9%E6%95%B0 HTTP/1.1&quot;</td>
</tr>
</tbody>
</table>

- The log management feature is currently only available in four regions including Beijing, Shanghai, Guangzhou, Chengdu and Toronto.
- The log management feature requires the source bucket and destination bucket to be in the same region.
- The destination bucket for storing the logs can be the source bucket itself, but this is not recommended.
- Currently, logs will be generated only when the bucket is accessed through XML APIs and XML API-based SDKs or tools, not via JSON APIs or JSON API-based SDKs or tools.
- Depending on customer needs and business development, COS may add new fields to the access logs. Please be sure to prepare for this when parsing the logs.

### Enabling Log Management

#### Using the Console

You can quickly enable the log management feature in the console. For instructions, see the Console Guide - Setting Log Management.

#### Using the API
To enable the log management feature for the specified bucket using API, follow the steps below:

1. Create a log role.
2. Grant the log role permissions.
3. Enable log management.

1. **Create a log role**

Create a log role. For more information, see [CreateRole](#).

Here, `roleName` must be `CLS_QcsRole`.

`policyDocument` must be:

```json
{
    "version": "2.0",
    "statement": [
        {
            "action": "name/sts:AssumeRole",
            "effect": "allow",
            "principal": {
                "service": "cls.cloud.tencent.com"
            }
        }
    ]
}
```

2. **Grant the log role permissions**

Grant the log role permissions. For specific API information, see [AttachRolePolicy](#).

Here, the `policyName` is `QcloudCOSAccessForCLSRole`; the `roleName` is `CLS_QcsRole` as in step 1, or, the `roleID` returned upon the creation of `roleName`, for this regard.

3. **Enable log management**

Call the API to enable the log management feature. For specific API information, see [PUT Bucket logging](#), where the destination bucket storing the logs is required to be in the same region as the source bucket.
Restrictions on Log Management

The log management feature is used to record the detailed access information of the specified source bucket and store the information in the specified bucket in the form of log files to facilitate bucket management.

Currently, the use limits of the log management feature are as follows:

- Delivery frequency limit: A log is generated every 5 minutes.
- Limit on the size of log file to be delivered: The log file to be delivered each time can be up to 256 MB. After this limit is exceeded, a new file will be delivered.
- Format of the log to be delivered: Each record is saved as one line and can contain multiple fields separated by space.
- Restrictions on the fields to be delivered: For more information on the fields to be delivered, see Log Management Overview.
- Invalid fields: If a field in a log contains the - character, the field is an invalid record or the default record.

Content Recorded in a Log

- The requests initiated by you to upload, download, or delete an object, to create or delete a bucket, or to modify the bucket configuration.
- The requests generated by content delivery through CDN and data pull from the COS origin server.

Content Not Recorded in a Log

- Offline origin-pull requests: If origin-pull is configured, if there are no objects in COS, the data will be downloaded from the origin server specified by you. This download operation is an offline origin-pull request and will not be recorded in the log.
- Redirects within a static website: If you configure redirection in the static website feature, you may be redirected to another page when accessing index.html. Such redirects will not be recorded in the log.
- Lifecycle transitions and deletions: If you configure the lifecycle to transition or delete objects after expiration, the operations of transitioning or deleting objects will be performed by the COS backend and will not be recorded in the log.
- Operations of listing objects and uploading inventory reports: The inventory feature automatically lists all or specified objects in your buckets and delivers the generated inventory report to your buckets on a periodical basis. The operations of listing objects and delivering inventory reports will not be recorded in the log.
- Operations of cross-region object replication: Cross-region replication involves getting an object from the source bucket and uploading it to the destination bucket. These operations are performed by the COS backend and will not be recorded in the log.
- Object downloads in the COS Select feature: The COS Select feature allows you to extract an object which needs to be obtained from a storage device first. Object downloads are performed by the COS backend and will not be recorded in the log.
Storing Tencent Cloud Product Logs in COS

Overview

Tencent Cloud products will generate a large number of logs during operations, which record your business situation for future analysis to facilitate your business development and decision-making. You can leverage the storage capabilities of COS to persistently store Tencent Cloud product logs. Specifically, you can use methods such as APIs, SDKs, and tools to easily and quickly get logs from COS and analyze them.

Storing Tencent Cloud product logs in COS helps meet your following needs:

- **Persistent storage**: COS provides stable and persistent storage services. You can persistently store your logs in COS at very low costs. When log-based analysis or decision-making is required for your business, you can get the logs for any time period from COS anywhere, anytime.
- **Data retrieval**: COS supports SELECT statements that provide simple retrieval and extraction features for logs stored in COS. You can retrieve desired information only by entering log fields, which can reduce the data download traffic.
- **Data analysis**: you can select one or multiple log files stored in COS, use Sparkling to analyze them, and make decisions based on the analysis results.

Log Shipping Methods

You can use either of the following two methods to store Tencent Cloud product logs to COS:

- Using the log shipping feature built in Tencent Cloud products; for example, for products such as COS, CLB, and CA, you can directly ship their logs to COS.
- Using the shipping feature of CLS: Tencent Cloud product logs that are shipped to CLS can be further shipped to COS through CLS for persistent storage.

Currently, the support of Tencent Cloud products for the two methods is as follows:

<table>
<thead>
<tr>
<th>Tencent Cloud Product Name</th>
<th>Directly Shipped to COS</th>
<th>Shipped to CLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CloudAudit (CA)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Cloud Load Balancer (CLB)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Cloud Kafka (CKafka)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>API Gateway</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Serverless Cloud Function (SCF)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Tencent Kubernetes Engine (TKE)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Live Video Broadcasting (LVB)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Tencent Cloud Base (TCB)</td>
<td>No</td>
<td>Yes; however, shipping from CLS to COS is not supported</td>
</tr>
<tr>
<td>Cloud Object Storage (COS)</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Directly shipping logs to COS

The following Tencent Cloud products support directly shipping logs to COS. You can configure log shipping rules as instructed in the product documentation to ship logs to COS.
### Cloud Object Storage

#### Log Shipping Document

<table>
<thead>
<tr>
<th>Tencent Cloud Product Name</th>
<th>Log Shipping Document</th>
<th>Log Shipping Interval</th>
<th>Log Shipping Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA</td>
<td>Click here</td>
<td>10–15 minutes</td>
<td>cloudaudit/customprefix/timestamp</td>
</tr>
<tr>
<td>CLB</td>
<td>Click here</td>
<td>60 minutes</td>
<td>lb-id/timestamp</td>
</tr>
<tr>
<td>CKafka</td>
<td>Click here</td>
<td>5–60 minutes, which can be specified</td>
<td>instance id/topic id/timestamp</td>
</tr>
<tr>
<td>COS</td>
<td>Click here</td>
<td>5 minutes</td>
<td>The path prefix can be specified. You are recommended to set a meaningful path, such as cos_bucketname_access_log/timestamp</td>
</tr>
</tbody>
</table>

Message data generated in CKafka can be shipped. If you want to get logs of operations such as CKafka instance creation, you can choose to ship CA logs.

### Shipping logs to COS through CLS

Certain Tencent Cloud products support log shipping to CLS for log retrieval and analysis. CLS can also ship logs to COS for persistent storage. For such products, you can enable log shipping to COS in the CLS Console to persistently store your log data. This reduces your storage costs and facilitates further offline analysis. Currently, the following products support log shipping to CLS:

<table>
<thead>
<tr>
<th>Tencent Cloud Product Name</th>
<th>Log Shipping Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>API Gateway</td>
<td>Click here</td>
</tr>
<tr>
<td>TKE</td>
<td>Click here</td>
</tr>
<tr>
<td>LVB</td>
<td>-</td>
</tr>
</tbody>
</table>

Logs in CLS can be shipped to COS by using any of the following three formats:

- Delimited format: you can ship log data in delimited format to COS. For more information, please see CSV Shipping.
- JSON format: you can ship log data in JSON format to COS. For more information, please see Shipping in JSON Format.
- Source format: you can ship log data in the source format to COS. Single-line and multi-line full text shipping is supported, and some source text can be shipped in CSV format. For more information, please see Shipping in Source Format.

You need to perform the following steps to ship logs to COS through CLS:

1. Select the product based on your business needs. Then, configure the logset and log topic as instructed in the product log shipping documentation provided above to import the log data generated by your business to CLS.
2. Select an appropriate format to ship data to COS based on your business needs. When shipping logs, you are recommended to enter the product name as the path prefix in order to differentiate between logs of different products. For example, you can name TKE logs as **TKE_tkeid_log/timestamp**.
3. After configuring a shipping rule, you can configure an event notification for file upload in SCF, so that you can perform further operations based on the event notification after logs are shipped to COS. For more information, please see Event Notifications.

### Log Analysis

**Downloading logs to local file system for offline analysis**
You can download logs to your local file system in various methods such as console, SDKs, APIs, or tools. Documentation for all download methods is provided below for your reference. You can replace the file path in the code with your actual log storage path to download logs.

<table>
<thead>
<tr>
<th>Download Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Click here</td>
</tr>
<tr>
<td>COSBrowser</td>
<td>Click here</td>
</tr>
<tr>
<td>COSCMD</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for Android</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for C</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for C++</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for .NET</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for Go</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for iOS</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for Java</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for JavaScript</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for Node.js</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for PHP</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for Python</td>
<td>Click here</td>
</tr>
<tr>
<td>SDK for WeChat Mini Program</td>
<td>Click here</td>
</tr>
<tr>
<td>API</td>
<td>Click here</td>
</tr>
</tbody>
</table>

### Analyzing logs with COS Select

You can use the COS Select feature to directly retrieve and analyze log files in CSV or JSON format stored in COS. With COS Select, you can filter desired log fields, which greatly reduces the data volume of logs transferred by COS and thus reduces your use costs and improves the data acquisition efficiency. For more information on COS Select, please see [Select Overview](#).

Currently, you can use COS Select through the console or API.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Click here</td>
</tr>
<tr>
<td>API</td>
<td>Click here</td>
</tr>
</tbody>
</table>
Overview

By enabling versioning, you can store multiple versions of an object in the same bucket. For example, you can store multiple objects with the same object key “picture.jpg” in one bucket, but the objects have different version IDs such as 100000, 100101, and 120002. You can query, delete, or restore the objects by version ID. Versioning enables you to recover from data loss caused by accidental deletion or application failure. Versioning is ideal for the following scenarios:

- If you need to delete objects (not permanently) in a versioning-enabled bucket, COS will insert delete markers for the deleted objects, and the markers will serve as the current object versions. You can restore an object to its previous version by the delete marker.
- If you need to replace objects, COS will insert new version IDs for the added objects, and you can still restore the replaced objects by version ID.

Versioning States

There are three versioning states for a bucket: not enabled, enabled, or suspended.

- **Versioning not enabled:** bucket versioning is not enabled by default.
- **Versioning enabled:** when bucket versioning is enabled, it will be applied to all the objects in the bucket. After versioning is enabled for the first time, objects added to the bucket thereafter will be assigned a unique version ID.
- **Versioning suspended:** Once the versioning state is changed from enabled to suspended (versioning cannot be disabled once enabled), objects added to the bucket thereafter will no longer be subject to versioning.

1. Once versioning is enabled for the bucket, it cannot be disabled. However, you can suspend versioning to stop object versioning.
2. For objects existing in the bucket before versioning is enabled, the version ID is “null”.
3. Enabling or suspending versioning will change the way COS makes requests when handling the objects but not the objects themselves.
4. Only the root account and authorized sub-accounts can suspend versioning for a bucket.

Managing Objects in Buckets with Different Versioning States

You can add, query, and delete objects in a bucket no matter what the versioning state is. When versioning is enabled or suspended, you can query and delete objects with or without version IDs.

- If versioning is not enabled, you can add, query, and delete objects in the same way as in an ordinary bucket. For more information, see the Object Management.
- If versioning is enabled or suspended, you can specify version IDs while retrieving and deleting objects. **Delete markers** will be added to deleted objects.

Managing Objects in a Versioning-enabled Bucket
For objects in the bucket before versioning is enabled, their version ID will be “null”. Enabling versioning will change the way COS handles the objects, such as how COS makes requests, but not the objects themselves. Objects with the same name added thereafter will be stored in the same bucket with different version IDs. You can manage objects in a bucket with versioning enabled as described below:

The way you add objects to a bucket is the same for both buckets with versioning enabled and not enabled, but the objects added to the two types of buckets have different version IDs. If versioning is enabled, COS will assign a unique version ID to any object added to the bucket; if versioning is not enabled, the version ID will remain null.

Adding Objects

If versioning is enabled for a bucket, when you perform PUT, POST, and COPY operations, COS will automatically assign a unique version ID to the objects added to the bucket.

As you can see below, when an object is added to a bucket with versioning enabled, COS will assign it a unique version ID.

Adding Objects

If versioning is enabled for a bucket, when you perform PUT, POST, and COPY operations, COS will automatically assign a unique version ID to the objects added to the bucket.

As you can see below, when an object is added to a bucket with versioning enabled, COS will assign it a unique version ID.

Listing Versioned Objects

COS stores object version information in the versions parameter associated with the bucket and returns object versions in the order of storage time from the newest to the oldest.

Retrieving All Versions of a Specific Object

You can retrieve all versions of a particular object using the versions parameter together with the prefix request parameter. For more information on prefix, see GET Bucket Object Versions.

Below is a sample request for retrieving all versions of a specific object:

```
GET /?versions&prefix=ObjectKey HTTP/1.1
```

Retrieving Historical Versions of an Object

A GET request with no version ID specified retrieves the current version of an object. The following figure shows how a GET request returns the current version (the latest version) of the 123.txt object.
A GET request with a version ID specified retrieves the specified version of an object. The following figure shows how a GET versionId request returns the specified version of the object. The request can also retrieve the current version.

Retrieving the Metadata of an Object Version

If you only want to retrieve the metadata of an object instead of its content, you can use the HEAD operation. By default, you get the metadata of the latest version. To retrieve the metadata of a specific object version, you need to specify its version ID when submitting the request.

To retrieve the metadata of an object version:

- Set versionId to the ID of the version of the object whose metadata you want to retrieve.
- Send a HEAD operation request with the versionId specified.

Deleting Objects

You can delete unnecessary object versions whenever you want. When you make a DELETE request in a versioning-enabled bucket, there will be two scenarios:

1. Perform a simple DELETE operation with no version ID specified.

   This is similar to putting the deleted objects to the "recycle bin". The objects are not permanently deleted and can be restored if needed.

   As you can see below, if you do not specify a version ID, the DELETE operation will not delete the 123.txt object; instead, it will
insert a delete marker and add a new version ID.

COS will insert a delete marker with a new version ID for the deleted objects. The delete marker will become the current version of the deleted objects. When you try to GET the objects with a delete marker, COS will assume that the objects do not exist and will return a 404 error.

2. If you delete an object with a version ID specified, the specified version of the object will be deleted permanently.

Delete Marker

A delete marker is a marker for versioned objects, indicating that the objects have already been deleted in COS. A delete marker has an object key and version ID like any other object. However, a delete marker differs from other objects in the following ways:

- Its content is empty.
- It does not have ACL value.
- If you try to GET a delete marker, you will get a 404 error.
- The only operation you can use on a delete marker is DELETE, and only the root account can issue such a request.

Deleting a Delete Marker

To delete a delete marker, you can specify its version ID in a DELETE Object versionId request to permanently delete it. If you use a DELETE request to delete a delete marker without specifying its version ID, COS will not delete the delete marker, but instead, insert another delete marker.
The following figure shows how a simple DELETE on a delete marker removes nothing, but adds a new delete marker to a bucket.

In a versioning-enabled bucket, a new delete marker has a unique version ID. Therefore, one object may have multiple delete markers in the same bucket. To delete a delete marker permanently, you must include its version ID in a DELETE Object versionId request.

The following figure shows how you can permanently delete a delete marker with a DELETE Object versionId request.

Only the root account can delete a delete marker permanently.

To permanently remove a delete marker:
1. Set versionId to the ID of the version of the delete marker you want to remove.
2. Send a DELETE Object versionId request.

Restoring Previous Versions

Versioning can be used to restore previous versions. You can do it in two ways:

1. Copy a previous version of the object into the same bucket
   The copied object will become the current version of the object, and all object versions will be retained
2. Delete the current version of the object permanently
   When you delete the current object version, you, in effect, turn the previous version into the current version of the object.

Managing Objects in a Versioning-suspended Bucket
When you suspend versioning, existing objects in your bucket do not change. What changes is how COS handles objects in future requests. The section below describes how to manage objects in a bucket where versioning is suspended.

**Adding Objects**

Once you suspend bucket versioning, when you use PUT, POST, or COPY operations, COS will automatically add a “null” version ID to objects added to the bucket thereafter as shown below:

If there are versioned objects in the bucket, the objects newly added to the bucket will become the current version with a version ID of null as shown below:

If there is already a null version in the bucket, it will be overwritten, and the original object content will also be replaced accordingly as shown below:

**Retrieving Historical Versions of an Object**

A GET Object request in a versioning-suspended bucket returns the current version of an object.

**Deleting Objects**

If versioning is suspended, a DELETE request will result in one of the following:

- If an object has a null version in the bucket, the null version of the object will be deleted.
  
  As you can see below, when you use a simple DELETE operation, COS will insert a delete marker for objects with a version ID of...
Since a delete marker does not have content, you will lose the original content of the null version when a delete marker replaces it.

- If an object does not have a null version in the bucket, a new delete marker will be added to the bucket. As you can see below, if an object does not have a null version in the bucket, a simple DELETE will not remove anything; COS will just insert a delete marker.

Even in a bucket where versioning is suspended, the root account is still able to delete a specified version permanently. The following figure shows that deleting a specified object version will delete the object permanently.
Only the root account can delete a specified object version permanently.
Use Cases

By enabling versioning, you can store multiple versions of an object in a bucket and retrieve, delete, or restore a specified version. For more information, see Versioning Overview.

Directions

**Configuring via the COS Console**

You can refer to Setting Versioning to learn how to encrypt objects in the console.

**Configuring via REST API**

You can use the REST API to configure versioning and manage objects in buckets with different versioning states. For more information, see the following API documentation:

- PUT Bucket versioning
- GET Bucket versioning
- GET Bucket Object versions
- PUT Object
- GET Object
- DELETE Object
- DELETE Multiple Objects

**Configuring with SDK**

You can directly call the Versioning method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- Android SDK
- C SDK
- C++ SDK
- .NET SDK
- Go SDK
- iOS SDK
- Java SDK
- JavaScript SDK
- PHP SDK
- Python SDK
Cross-region Replication

Overview

With a cross-region replication rule, COS can automatically and asynchronously replicate the new objects added to one bucket to another bucket in a different region. When cross-region replication is enabled, COS will accurately replicate the object content (such as object metadata and version ID) in the source bucket to the destination bucket, and the object copies contain exactly the same attribute information. Additionally, operations on the objects in the source bucket, such as adding or deleting, will also be synced to the destination bucket.

To enable cross-region replication, you need to make sure that the source and destination buckets are in different regions and both have versioning enabled.

After cross-region replication is enabled, the object copies will be of the same storage class as the source objects, unless you specify a different storage class for the object copies during replication.

During replication, COS will copy the access control list (ACL) of the source bucket. Currently, the source bucket and destination bucket must be in the same account.

Applicable Scenarios

- **Remote disaster recovery**: COS boasts 11 nines of availability for object data, but there is still a slight chance of data loss due to force majeure such as wars and natural disasters. If you want to avoid data loss by explicitly having a separate copy in a different region, you can use cross-region replication to achieve remote disaster recovery. In this way, when the IDC in one region is damaged due to force majeure, the IDC in the other region can still provide data copies for your use.

- **Compliance**: COS ensures data availability by providing multiple copies and erasure codes for data in physical disks by default. However, there may be compliance requirements in some industries stipulating that you keep copies in another region. Cross-region replication allows data to be replicated across regions to meet such requirements.

- **Access latency reduction**: when you have end users accessing objects from different regions, with cross-region replication, you can maintain object copies in the available storage regions closest to them, which can minimize access latency to deliver a better user experience.

- **Special technical requirements**: if you have computing clusters in two different regions and the clusters need to process the same set of data, with cross-region replication, you can maintain object copies in both regions.

- **Data migration and backup**: You can copy your business data from one availability region to another one as needed to implement data migration and backup.

Notes

**Replication Time**

The time it takes for COS to replicate objects depends on factors such as object size, distance between storage regions, and how the objects were uploaded. The time needed for synchronization may range from a few minutes to a few hours, subject to such factors.

- **Object size**: it takes more time to replicate large objects, for which multipart upload is recommended for faster upload and synchronization.
Distance between the source and destination regions: the farther the regions are away from each other, the longer it takes to transfer data.

- How the objects were uploaded: unlike simple upload where data can only be uploaded or downloaded serially, multipart upload supports concurrent uploads which can speed up upload and cross-region replication of large files. For more information, see Simple Upload and Multipart Upload.

Lifecycle

To use cross-region replication, you need to enable versioning, which will keep multiple historical versions of objects in the buckets and result in more storage consumption. COS cross-region replication will incur fees for data requests, downstream traffic, and data storage. Among them, data storage is charged at the prices in the destination bucket region. If you want to reduce the costs or customize your data retention approach, you can do so through lifecycle management based on your business needs.

- If you want the object copies in the destination bucket to follow the same lifecycle rule as those in the source bucket, create the same lifecycle rule in the destination bucket.
- If you have set a lifecycle rule for the destination bucket, note that the creation time of the object copies generated by cross-region replication is the creation time of the source objects but not the time when the copies are replicated to the destination bucket.
- If you have set a lifecycle rule in the source bucket and an object being replicated needs to be deleted according to the lifecycle rule, the object will still be replicated, and the object copy will be retained in the destination bucket.

Versioning

To use cross-region replication, you need to enable versioning for both the source and destination buckets. For more information, see Versioning Overview. Once versioning is enabled, you should note that disabling it will affect cross-region replication:

- If you try to disable versioning in a bucket where cross-region replication is enabled, COS will return an error prompting you to delete the cross-region replication rule before disabling versioning.
- If you try to disable versioning in a destination bucket, COS will prompt you that cross-region replication will be affected if you do so. If you proceed to disable versioning, the cross-region replication rule that uses this bucket as the destination bucket will become invalid.
Replication Behavior Descriptions


This document describes the content that will and will not be copied after you enable cross-region replication for a bucket.

Content That Will Be Copied

In a source bucket with cross-region replication enabled, COS will copy the following:

- Any new objects uploaded to the source bucket after the cross-region replication rule is added.
- Object attribute information such as object metadata and version ID.
- Information on object operations such as adding an object of the same name (equivalent to adding a new object) and deleting an object.

- If you specify to delete an object version from the source bucket (i.e., specifying the version ID), this operation will not be copied.
- If you add a bucket-level configuration such as a lifecycle rule to the source bucket, object operations caused by such configurations will not be copied to the destination bucket.

Deletions with Cross-region Replication Enabled

If an object is deleted from a source bucket with cross-region replication enabled, the following will happen:

- When a DELETE request is executed with no object version ID specified, COS will add a delete marker to the source bucket, and cross-region replication will copy the marker to the destination bucket. For more information on versioning and delete marker, see Versioning Configuration.
- When a DELETE request is executed with an object version ID specified, COS will delete the specified object version from the source bucket but will not copy the deletion operation to the destination bucket, which means that COS will not delete the specified object version from the destination bucket. This can prevent malicious deletion of data.

Contents That Will Not Be Copied

In a source bucket with cross-region replication enabled, COS will not copy the following:

- Objects that have already been in the bucket when cross-region replication is enabled, i.e., the legacy data.
- Encrypted objects. COS does not copy encrypted data by default, but you can enable encrypted data replication by modifying the configuration information.
- New object data added to the source bucket that is copied from another bucket.
- Configuration updates at the bucket level.
- Results from the execution of the lifecycle configuration.

- The cross-region replication of object data between buckets is not transitive. If you set two cross-region replication rules, one of which sets bucket A as the source bucket and bucket B as the destination bucket, and the other sets bucket B as the source bucket and bucket C as the destination bucket, then the object data added to bucket A will only be copied to bucket B but not to bucket C.
- For example, if you update the lifecycle configuration of the source bucket, COS will not apply this configuration to the destination bucket synchronously.
If you only configure a lifecycle rule for the source bucket, COS will add delete markers to expired objects there, but the markers will not be copied to the destination bucket. If you want the destination bucket to delete the expired objects, you need to configure the same lifecycle rule for the destination bucket separately.
Cross-region Replication Configuration


Applicable Scenarios

With cross-region replication enabled, object data in your source bucket can be copied to the designated destination bucket in another region. Cross-region replication can help you achieve remote disaster recovery, comply with industry-specific requirements, migrate and back up data, reduce access latency, enable clusters in different regions to access data, etc.

After versioning is enabled, newly uploaded objects will generate multiple versions and take up storage space, so these versions of the object will also charge for storage.

Directions

Configuring via the COS Console

For information on how to configure a cross-region replication rule in the COS Console, see Setting Cross-region Replication documentation.

Configuring via REST API

You can configure and manage cross-region replication via REST API as described in the following API documentation:

- PUT Bucket replication
- GET Bucket replication
- DELETE Bucket replication

Configuring with SDK

You can directly call the cross-region replication method in the SDK. For more information, see the SDK documentation for the corresponding programming language below:

- Android SDK
- C SDK
- C++ SDK
- .NET SDK
- Go SDK
- iOS SDK
- Java SDK
- JavaScript SDK
- PHP SDK
- Python SDK
Overview of Multi-AZ Feature

Multi-AZ (availability zone) refers to the multi-AZ storage architecture offered by Tencent Cloud COS, which can provide IDC-level disaster recovery capabilities for your data.

Your data is scattered among multiple IDCs in a region. When an IDC fails due to extreme situations such as natural disasters or power failures, the multi-AZ storage architecture can still provide stable and reliable storage services.

Multi-AZ provides 99.9999999999% (12 nines) design data reliability and 99.995% design service availability. When you upload data objects to COS, you can store them in a multi-AZ region simply by specifying the storage class.

Currently, the multi-AZ feature of COS is supported only in the Guangzhou region and will be available in other public cloud regions in the future. Multi-AZ storage is more costly. For more information, please see Product Pricing.

Strengths of Multi-AZ

If you store your data in a multi-AZ region, the data will be divided into multiple chunks, and corresponding coding chunks will be calculated based on the erasure code algorithm. The original data chunks and coding chunks will be mixed up and evenly distributed to different IDCs in the region for storage and intra-region disaster recovery. When an IDC becomes unavailable, the data can still be read from or written to other IDCs normally, ensuring that your data can be persistently stored without loss, which maintains your business data continuity and high availability. The COS multi-AZ feature has the following strengths:

- **Intra-region disaster recovery**: cross-IDC disaster recovery is supported. In the multi-AZ storage architecture, object data is stored on different devices in different IDCs in the same region. When an IDC fails, other redundant IDCs remain available, guaranteeing that your business will not be affected, and data will not be lost.
- **Stability and durability**: the erasure code-based redundant storage mechanism is leveraged to provide up to 99.9999999999% design data reliability. Data is stored in chunks and read and written concurrently to provide up to 99.995% design service availability.
- **Ease of use**: you can specify the storage architecture for your data by specifying the object storage class. You can even select any objects in a bucket and store them in the multi-AZ architecture for greater ease of use.

Multi-AZ storage and non-multi-AZ storage are compared below for their specifications and limitations:

<table>
<thead>
<tr>
<th>Item</th>
<th>Multi-AZ Storage</th>
<th>Non-Multi-AZ Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design data persistence</td>
<td>99.99999999999% (12 nines)</td>
<td>99.9999999999% (11 nines)</td>
</tr>
<tr>
<td>Design service availability</td>
<td>99.995%</td>
<td>99.99%</td>
</tr>
<tr>
<td>Supported region</td>
<td>Guangzhou</td>
<td>All regions. For more information, please see Regions and Access Domain Names</td>
</tr>
<tr>
<td>Supported storage class</td>
<td>Standard (MAZ-STANDARD)</td>
<td>Standard (STANDARD), Standard_IA (STANDARD-IA), and Archive (ARCHIVE)</td>
</tr>
</tbody>
</table>

Directions

©2013-2019 Tencent Cloud. All rights reserved.
You can enable multi-AZ for a bucket and set the object storage class to multi-AZ for objects uploaded to it. Currently, the multi-AZ storage architecture supports only the Standard storage class. When uploading an object, you can store it in the multi-AZ storage architecture by simply specifying the object storage class.

In short, you only need to perform the following two steps to store files in the multi-AZ architecture:

1. Create a bucket and enable multi-AZ during creation. For more information on how to create a bucket, please see Creating Buckets.
2. Upload a file and specify the file storage class during upload. For more information on how to upload a file, please see Uploading Objects.

- Once multi-AZ is enabled for the bucket, it cannot be disabled, so please enable it with caution. Multi-AZ is not enabled for existing buckets by default, and it can be enabled only for new buckets.
- For buckets with multi-AZ enabled, objects can only be uploaded in the Multi-AZ Standard (MAZ_STANDARD) storage class. The Multi-AZ Standard_IA (MAZ_STANDARD_IA) and Multi-AZ Archive (MAZ_ARCHIVE) storage classes will be supported in the future.
- If you want to store existing data to a multi-AZ bucket, you can create a bucket with multi-AZ enabled and use the batch replication feature of COS Batch to replicate files in the existing bucket to the new bucket in batches. For more information on how to use COS Batch, please see Batch Operation.

Use Limits

Currently, with the multi-AZ feature of COS, objects can only be uploaded in the Multi-AZ Standard storage class (MAZ_STANDARD). Therefore, there are limits on features related to storage class change as detailed below:

- Storage class limit: currently, only the Multi-AZ Standard (MAZ_STANDARD) storage class is supported. The Multi-AZ Standard_IA (MAZ_STANDARD_IA) and Multi-AZ Archive (MAZ_ARCHIVE) storage classes will be supported in the future.
- Operation limit: currently, objects can only be uploaded, downloaded, and deleted. Objects can be replicated to a multi-AZ bucket but not to a single-AZ bucket.
- Lifecycle limit: currently, expired objects can only be deleted but not transitioned to other storage classes.
- Cross-region replication limit: currently, multi-AZ is only supported in the Guangzhou region; therefore, cross-region replication is not supported.
Data Security
Server-side Encryption Overview

Overview

COS encrypts your data at the object level before it is written to IDC disks and automatically decrypts the data when you access it. Encryption and decryption are completed on servers. Server-side encryption can effectively protect static data.

- As long as you have access permission to objects, user experience is the same accessing encrypted and non-encrypted objects.
- Server-side encryption only encrypts object data but not its metadata. Server-side encrypted objects can only be accessed with a valid signature, not by anonymous users.

Use Cases

- **Private data storage**: For private data storage, server-side encryption can encrypt stored data to protect your privacy and automatically decrypt the data when you access it.
- **Private data transfer**: For private data transfer, COS supports deploying SSL certificates with HTTPS to implement encryption. An encryption layer will be established on the transfer linkage layer, ensuring that data will not be stolen or tampered with during transfer.

Encryption

COS supports multiple server-side encryption methods such as SSE-COS and SSE-C. You can choose the appropriate one to encrypt data stored in COS.

**SSE-COS Encryption**

SSE-COS: Server-side encryption with a key managed by COS. In this mode, COS will manage the master key and data, and users can manage and encrypt the data directly through COS. SSE-COS uses a strong AES-256 multi-factor encryption to ensure that each object is encrypted with a unique key, while regularly rotating the master key to encrypt the key itself.

- When uploading an object using the **POST** operation, you need to provide the same information in the form field. Instead of providing the `x-cos-server-side-encryption-*` header. For more information, see [POST Object](#).
- SSE-COS encryption is not available for objects uploaded with a pre-signed URL. Instead, you need to use COS Console or HTTP request header to specify server-side encryption.

Using the COS Console

You can refer to [Setting Object Encryption](#) to learn how to encrypt objects in the console.

Using REST API

- When you list objects in a bucket, all objects will be listed, no matter whether they are encrypted.
When uploading an object using the POST operation, please provide the same information in the form field, instead of providing the request header. For more information, see POST Object.

When you request the following APIs, you can apply server-side encryption by providing the x-cos-server-side-encryption header. For more information, see Common Request Headers - SSE-COS.

- PUT Object
- Initiate Multipart Upload
- PUT Object - Copy
- POST Object

**SSE-COS Encryption**

SSE-KMS encryption is server-side encryption using a key managed by KMS. KMS is a security management service launched by Tencent Cloud, using a third-party-certified hardware security module (HSM) to generate and protect keys. KMS allows users to easily create and manage keys, meeting their key management needs for multiple applications and services, while satisfying regulatory and compliance requirements.

When using SSE-KMS encryption for the first time, you need to enable KMS service. After KMS service is enabled, the system will automatically create a default customer master key (CMK) for you. You can also create your own keys through KMS Console, define key policies and use methods, KMS allows users to choose their own key material from KMS or external sources, see Create Key and Import External Key.

- SSE-KMS only encrypts the object data, not its metadata.
- Currently, SSE-KMS only supports Beijing, Shanghai, and Guangzhou regions.
- Using SSE-KMS encryption will incur additional cost, which will be charged by KMS, please see KMS Billing Overview for details.

**Using the COS Console**

You can refer to Setting Object Encryption to learn how to encrypt objects in the console.

**Using REST API**

- When you list objects in a bucket, all objects will be listed, no matter whether they are encrypted.
- When uploading an object using the POST operation, please provide the same information in the form field, instead of providing the request header. For more information, see POST Object.

When you request the following APIs, you can apply server-side encryption by providing the x-cos-server-side-encryption header. For more information, see Common Request Headers - SSE-COS.

- PUT Object
- Initiate Multipart Upload
- PUT Object - Copy
- POST Object

**Notes**

If you have never used COS console for SSE-KMS encryption, and only used API for SSE-KMS encryption, you need to create CAM Role first. The specific creation steps are as follows:

1. Log into the CAM Console and go to the Roles page.
2. Click [Create Role] and select the role entity as [Tencent Cloud Product Service].
3. Select services supporting roles as [Object Storage], and then click [Next].

4. Configure the role policy, find and select [QcloudKMSCreaterFullAccess], and then click [Next].
5. Enter the specified role name: COS_QcsRole.

6. Click [Done] to complete the creation.

**SSE-C Encryption**

SSE-C encryption is server-side encryption with a user-defined key. When you upload an object, COS will use the encryption key you provide to apply AES-256 encryption to your data.

- COS does not store your encryption key. Instead, it stores the HMAC value of the encryption key with random data added, which is used to verify your request to access the object. COS cannot use the HMAC value to derive the value of the encryption key or decrypt the encrypted object. Therefore, if you lose the encryption key, the object cannot be obtained again.
- When uploading an object using the **POST** operation, you need to provide the same information in the form field, instead of providing the **x-cos-server-side-encryption-** header. For more information, see **POST Object**.
- SSE-C can only be used via APIs but not the console.

**Using REST API**

When you request the following APIs, you can apply server-side encryption to **PUT** and **POST** requests by providing the **x-cos-server-side-encryption-** header. For **GET** and **HEAD** requests, you need to provide the **x-cos-server-side-encryption-** header to decrypt the specified object encrypted with SSE-C. For more information, see **Common Request Headers - SSE-C**. The following operations support this header:

- **GET Object**
- **HEAD Object**
- **PUT Object**
- **Initiate Multipart Upload**
- **Upload Part**
- POST Object
- PUT Object - Copy
Bucket Encryption Overview

Overview

Once a bucket is configured for encryption, all objects uploaded to the bucket thereafter will be encrypted by default with the specified encryption method.

Currently supported encryption methods include:

- SSE-COS encryption: server-side encryption using COS-managed encryption keys

For more information on server-side encryption, please see Server-side Encryption Overview.

Directions

Via the COS Console

You can set bucket encryption in the COS Console. For more information, please see Setting Bucket Encryption in Console Guide.

Via REST API

You can configure bucket encryption using the following APIs:

- PUT Bucket encryption
- GET Bucket encryption
- DELETE Bucket encryption

Notes

Uploading objects to an encrypted bucket

For buckets requiring encryption, note the following:

- Configuring encryption for a bucket will not lead to encryption operations on objects that already exist in it.
- After encryption is configured for a bucket, for objects uploaded to the bucket:
  - If your PUT request does not contain encryption information, the uploaded objects will be encrypted with the encryption configuration of the bucket.
  - If your PUT request contains encryption information, the uploaded objects will be encrypted with the contained encryption information.
- After encryption is configured for a bucket, for inventory reports delivered to the bucket:
  - If encryption is not configured for the inventory, the delivered reports will be encrypted with the encryption configuration of the bucket.
  - If encryption is configured for the inventory, the delivered reports will be encrypted with the encryption configuration of the inventory.
- After encryption is configured for the bucket, the data pulled back to the bucket will be encrypted with the encryption configuration of the bucket by default.

Encrypting buckets that have cross-region replication rules configured

For the destination bucket that has a cross-region replication rule configured, if you configure encryption for it, note the following:

- If the objects in the source bucket are not encrypted, the object copies in the destination bucket will be encrypted by default.
- If the objects in the source bucket are encrypted, the object copies in the destination bucket will inherit the encryption from the source bucket, and the bucket encryption settings will not be honored.
Cloud Access Management (CAM)

Access Control

Basic Concepts of Access Control

Last updated: 2020-02-26 14:26:04

Concept

You can grant access permissions by specifying a person to perform a specified action on specified resources under a specified condition. Generally, the following four elements are used to describe an access permission action: identity, resource, action, and condition (optional).

Elements for Access Permissions

Tencent Cloud’s identity

When you apply for a Tencent Cloud account, the system will create a root account for logging in to the Tencent Cloud services. The Tencent Cloud root account manages different types of users with different roles using the user management feature. User types include collaborator, message recipient, sub-user, and role. For more information, see CAM Identity Management and Glossary.

COS resources

Bucket and Object are the basic resources of the COS service, and they have subresources associated with them.

Bucket’s subresources include:
- acl and policy: access control information of a bucket
- website: static website hosting configuration of a bucket
- tagging: tag information of a bucket
- cors: cross-origin configuration of a bucket
- lifecycle: lifecycle configuration of a bucket

Object’s subresources include:
- acl: access control information of an object
- restore: restore configuration of an archive object

COS operations

COS provides a range of API operations on various resources. For more information, see Operation List.

CAM Overview

Private principle

Note:
By default, all Tencent Cloud COS resources are private.
The resource owner (the Tencent Cloud root account creating a bucket resource) has the highest permission on the resource, and can grant access permissions to others or anonymous users by editing and writing an access policy.

When a Tencent Cloud CAM account is used to create a bucket or upload an object, its parent root account is the resource owner.

The root account of the bucket owner can authorize other Tencent Cloud root accounts to upload objects (i.e. cross-account upload). In this case, the object owner is still the root account of the bucket owner.

### Access control policy

#### Resource-based policy

Tencent Cloud COS supports access control at both **bucket** and **object** dimensions, as detailed below:

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Type</th>
<th>Language</th>
<th>Supported Identity</th>
<th>Supported Resource Granularity</th>
<th>Supported Action</th>
<th>Supported Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bucket</td>
<td>Access policy language (Policy)</td>
<td>JSON</td>
<td>Sub-accounts, roles, Tencent Cloud services, other root accounts, anonymous users, etc.</td>
<td>Buckets, objects, prefixes, etc.</td>
<td>Each specific action</td>
<td>Allow/Deny</td>
</tr>
<tr>
<td>Bucket</td>
<td>Access Control List (ACL)</td>
<td>XML</td>
<td>Other root accounts, anonymous users, etc.</td>
<td>Buckets</td>
<td>Read and write actions</td>
<td>Allow</td>
</tr>
<tr>
<td>Object</td>
<td>Access Control List (ACL)</td>
<td>XML</td>
<td>Other root accounts, anonymous users, etc.</td>
<td>Objects</td>
<td>Read and write actions</td>
<td>Allow</td>
</tr>
</tbody>
</table>

### Bucket Policy

A bucket policy is described in JSON language, and supports granting anonymous identities or any Tencent Cloud CAM account the permissions to access and perform operations on buckets and objects. In Tencent Cloud COS, the bucket policy can be used to manage almost all operations in the bucket. It is recommended that you use a bucket policy to manage access policies that cannot be described using ACLs.

A Tencent Cloud root account has the highest permission on its resources (including buckets). Even if you can set limits on almost all operations in the bucket policy, the root account always has the permission for the PUT Bucket Policy operation, and can call this operation without checking the bucket policy.

The following policy allows anonymous users to access all objects in the bucket `examplebucket-1250000000` in Guangzhou, and to download all objects (GetObject) in the bucket without signature verification. In this case, any anonymous user who knows the URL can download the objects (similar to public-read):

```json
{
    "Statement": [
        {
            "Principal": "*",
            "Effect": "Allow",
            "Action": ["cos:GetObject"],
            "Resource": ["qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"]
        }
    ],
    "Version": "2.0"
}
```
ACL

An ACL is described in the XML language. It is a list of specified grantees and permissions granted, which is associated with resources. Each bucket and object has an associated ACL to grant basic read and write permissions to anonymous users or other Tencent Clouds root accounts.

The bucket ACL in this example describes the full control permission of the bucket owner (UIN: 100000000001):

```xml
<AccessControlPolicy>
  <Owner>
    <ID>qcs::cam::uin/100000000001:uin/100000000001</ID>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="RootAccount">
        <ID>qcs::cam::uin/100000000001:uin/100000000001</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

The object ACL in this example describes the full control permission of the object owner (UIN: 100000000001) and grants the read permission to all users (the public-read permission to anonymous users):

```xml
<AccessControlPolicy>
  <Owner>
    <ID>qcs::cam::uin/100000000001:uin/100000000001</ID>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>qcs::cam::uin/100000000001:uin/100000000001</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee URI="http://cam.qcloud.com/groups/global/AllUsers">
        <ID>qcs::cam::uin/100000000001:uin/100000000001</ID>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

User-based policy

In CAM, you can grant different permissions to different types of users under the root account.

The biggest difference between a user policy and a bucket policy is that the user policy only describes the effect, the action, the resource, and the condition (optional), and does not describe the identity (principal). Therefore, after you write a user policy, associate it with a sub-user, a user group or a role. Besides, the user policy does not support granting anonymous users the permissions to access resources and perform operations.
You can **associate a preset policy for authorization**, or **write a user policy** and associate it with a specified identity to manage access for your users.

The following policy example grants the permission to perform all COS operations on the bucket `examplebucket-1250000000` in Guangzhou. You need to save the policy and then associate it with a CAM sub-user, a user group or a role before it takes effect.

```json
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["cos:*"],
            "Resource": [
                "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*",
                "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/"
            ]
        }
    ],
    "Version": "2.0"
}
```
Notes on Principle of Least Privilege

Last updated: 2020-01-15 15:25:46

Overview

When using COS, you may need to use a temporary key to grant users permissions to certain resources or operations, configure user policies for your sub-users or collaborators that allow them to help you operate on the resources in COS, or create bucket policies that allow the specified users to perform certain operations on or access certain resources in your bucket. When configuring these permissions, please comply with the principle of least privilege in order to ensure the security of your data assets.

The principle of least privilege means that when granting permission, you must specify the scope of the permission granted to the specified user for performing what operation and access what resource under what conditions.

Precautions

You are recommended to strictly comply with the principle of least privilege to ensure that a user can only perform the specified operations (e.g., action:GetObject) or access the specified resources (e.g., resource:exampleBucket-APPID/exampleObject.txt).

To prevent data security risks caused by unexpected and unauthorized operations with excessive permissions, it is strongly recommended that you avoid authorizing a user to access all resources (e.g., resource:* ) or perform all operations (e.g., action:* ).

Below are some potential data security risks:

- **Data leakage**: if you want to authorize a user to download the specified resources such as examplebucket-1250000000/data/config.json and examplebucket-1250000000/video/ but include examplebucket-1250000000/* in the permission policy, then all objects in the bucket can be downloaded without your authorization, leading to unexpected data leakage.

- **Data overwriting**: if you want to authorize a user to upload examplebucket-1250000000/data/config.json and examplebucket-1250000000/video/ but include examplebucket-125000000/* in the permission policy, then all objects in the bucket can be uploaded without your authorization, which may overwrite unintended objects. To avoid this risk, in addition to following the principle of least privilege, you can retain all versions of data for traceability as instructed in Versioning Overview.

- **Permission leakage**: if you want to authorize a user to list the objects in the bucket (cos:GetBucket) but configured cos:* in the permission policy, then all operations on the bucket will be allowed, including reauthorizing the bucket, deleting objects, and deleting the bucket, which puts your data at extremely high risk.

Usage Guide

Under the principle of least privilege, you should specify the following information in the policy:

- **principal**: you should specify to which sub-account (user ID required), collaborator (user ID required), anonymous user, or user group to grant permission. This is not needed if you use a temporary key for access.

- **statement**: enter the corresponding parameters.
  - **effect**: you must specify whether the policy is to "allow" or "deny".
  - **action**: you must specify the allowed or denied operation. An operation can be an API (described using the prefix "name") or a feature set (a set of specific APIs, described using the prefix "permid").
  - **resource**: you must specify the resource authorized by the policy. A resource is described in a six-piece format. You can set the resource scope as the specified file (e.g., exampleobject.jpg ) or the specified directory (e.g., examplePrefix/* ). Unless it is required by your business, please do not grant any user access to all resources using the * wildcard.
  - **condition**: it describes the condition for the policy to take effect. A condition consists of operator, action key, and action value. A condition value may contain information such as time and IP address.

Least privilege guide for temporary key
When applying for a temporary key, you can set the permission policy field [Policy] to limit the permissions to operations and resources within the specific scope. For more information, please see Generating and Using Temporary Keys.

Authorization example

Granting a user permission to access the specified object using the SDK for Java

If you want to use the SDK for Java to grant a user permission to download the exampleObject.txt object in the examplebucket-125000000 bucket, the configuration code should be as follows:

```java
// Import `java sts sdk` using the integration method with Maven as described on GitHub
import java.util.*;
import org.json.JSONObject;
import com.tencent.cloud.CosStsClient;

class Demo {
    public static void main(String[] args) {
        TreeMap<String, Object> config = new TreeMap<String, Object>();
        try {
            // Replace with your own SecretId
            config.put("SecretId", "AKIDHTVVaVR6e3");
            // Replace with your own SecretKey
            config.put("SecretKey", "PdkhT9e2rZCfy6");

            // Validity period of the temporary key in seconds; default value: 1,800s; maximum value: 7,200s
            config.put("durationSeconds", 1800);

            // Replace with your own bucket
            config.put("bucket", "examplebucket-125000000");
            // Replace with the region where your bucket resides
            config.put("region", "ap-guangzhou");

            // Change it to the allowed path prefix such as `a.jpg`, `a/`, or `/`. You can determine the specific path to which files can be uploaded based on your login status
            // If "*" is entered, the user will be allowed to access all resources; unless it is required by your business, please grant the user only the corresponding permission based on the principle of least privilege.
            config.put("allowPrefix", "exampleObject.txt");

            // List of key permissions. The following permissions are required for simple upload, upload using a form, and multipart upload. For other permissions, please visit https://cloud.tencent.com/document/product/436/31923
            String[] allowActions = new String[] {
                // Download data
                "name/cos:GetObject"
            };
            config.put("allowActions", allowActions);

            JSONObject credential = CosStsClient.getCredential(config);
            // If it succeeds, the temporary key information will be returned and printed out as shown below
            System.out.println(credential);
        } catch (Exception e) {
            // If it fails, an exception will be thrown
            throw new IllegalArgumentException("no valid secret !");
        }
    }
}
```

Granting a user permission to access the specified object using API

If you want to use an API to grant a user permission to download the exampleObject.txt object in the examplebucket-125000000 bucket and all objects in the examplePrefix directory, the access policy should be as follows:
A user policy is a user permission policy created in the CAM Console to grant a user permission to access certain resources in COS. For more information, please see Access Policy Language Overview.

Authorization example

Granting an account permission to access the specified object

If you want to grant an account whose UIN is 100000000001 permission to download the exampleObject.txt object in the examplebucket-1250000000 bucket, the access policy should be as follows:

```json
{
  "version": "2.0",
  "principal": {
    "qcs": [
      "qcs::cam::uin/100000000001:uin/100000000001"
    ]
  },
  "statement": [
    {
      "action": ["name/cos:GetObject"],
      "effect": "allow",
      "resource": [
        "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000:ap-guangzhou.myqcloud.com/exampleObject.txt"
      ]
    }
  ]
}
```

Granting a sub-account permission to access the specified directory

If you want to grant a sub-account whose UIN is 100000000011 (root account UNI: 100000000001) permission to download the objects in the examplePrefix directory in the examplebucket-1250000000 bucket, the access policy should be as follows:

```json
{
  "version": "2.0",
  "principal": {
    "qcs": [
      "qcs::cam::uin/100000000001:uin/100000000001"
    ]
  },
  "statement": [
    {
      "action": ["name/cos:GetObject"],
      "effect": "allow",
      "resource": [
        "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000:examplePrefix/*";
      "qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000:examplePrefix/*"
      ]
    }
  ]
}
```
Least privilege guide for bucket policy

A bucket policy is an access policy configured for a bucket to allow the specified user to perform certain operations on the bucket and resources in it. For more information, please see Adding Bucket Policies.

Authorization example

Granting a sub-account permission to access the specified objects

If you want to grant a sub-account whose UIN is 100000000011 (root account UNI: 100000000001) permission to download the exampleObject.txt object in the examplebucket-1250000000 bucket and all objects in the examplePrefix directory, the access policy should be as follows:

```json
{
  "Statement": [
    {
      "Action": [
        "name/cos:GetObject"
      ],
      "Effect": "allow",
      "Principal": {
        "qcs": [
          "qcs::cam::uin/10000000001:uin/100000000011"
        ]
      },
      "Resource": [
        "qcs::cos:ap-beijing:uid/1250000000:examplebucket-1250000000/exampleObject.txt",
        "qcs::cos:ap-beijing:uid/1250000000:examplebucket-1250000000/examplePrefix/*"
      ]
    }
  ],
  "version": "2.0"
}
```
Policy-based Access Control

Access Policy Language Overview

Overview

An access policy which employs the JSON-based access policy language is used to grant the access to COS resources. You can authorize a specified principal to perform actions on a specified COS resource through the access policy language.

The language describes the basic elements and usage of a bucket policy. For more information, see CAM Policy Management.

Elements in an Access Policy

The access policy language contains the following basic elements:

- **Principal**: describes the entity to be authorized by the policy. This includes users (root accounts, sub-accounts, anonymous users), and user groups. This element can only be used in bucket access policies, rather than user access policies.

- **Statement**: describes the detailed information of one or more permissions. This element includes a permission or a permission set of multiple elements such as effect, action, resource, and condition. A policy has only one statement.
  - **Effect**: Describes whether the result of the statement is "allowed" or "explicitly denied". This includes two situations: allow and deny. This element is required.
  - **Action**: Describes allowed or denied actions. An action can be an API (described using the prefix "name") or a feature set (a set of specific APIs, described using the prefix "permid"). This element is required.
  - **Resource**: Describes the detailed data of authorization. A resource is described in a six-piece format. Detailed resource definitions vary by product. For more information on how to specify a resource, see the documentation for the product whose resources you are writing a statement for. This element is required.
  - **Condition**: Describes the condition for the policy to take effect. A condition consists of operator, action key, and action value. A condition value may contain information such as time and IP address. Some services allow you to specify additional values in conditions. This element is optional.

How to Use Elements

**Specifying a Principal**

A principal is used to specify the user, account, service, or entity that is allowed or disallowed to access resources. It only works in buckets and is not required in user policies because these policies are directly added to the specific user. The following example specifies a principal.

```json
"principal": {
  "qcs": [
  "qcs::cam::uin/100000000001:uin/100000000001"
  ]
}
```

Grant permissions to an anonymous user:

```json
"principal": {
  "qcs": [
  "qcs::cam::anonymous:anonymous"
  ]
}
```
Grant permissions to a root account (UIN: 100000000001):

```
"principal": {
  "qcs": [
    "qcs::cam::uin/100000000001:uin/100000000001"
  ]
}
```

Grant permissions to a sub-account (UIN: 100000000011) under the root account (UIN: 100000000001):

```
"principal": {
  "qcs": [
    "qcs::cam::uin/100000000001:uin/100000000011"
  ]
}
```

**Specifying an Effect**

If you don't explicitly grant access to (allow) a resource, the access is implicitly denied. You can also explicitly deny access to a resource, which you might do to make sure that a user cannot access it, even if a different policy grants it access. The following example specifies an "allow" effect.

```
"effect": "allow"
```

**Specifying an Action**

COS defines COS actions that you can specify in a policy. The specified actions are just like initiation of API requests.

### Actions on Buckets

<table>
<thead>
<tr>
<th>Description</th>
<th>API</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>name/cos:GetService</code></td>
<td>GET Service</td>
</tr>
<tr>
<td><code>name/cos:GetBucket</code></td>
<td>GET Bucket (List Object)</td>
</tr>
<tr>
<td><code>name/cos:PutBucket</code></td>
<td>PUT Bucket</td>
</tr>
<tr>
<td><code>name/cos:DeleteBucket</code></td>
<td>DELETE Bucket</td>
</tr>
</tbody>
</table>

### Actions on Objects

<table>
<thead>
<tr>
<th>Description</th>
<th>API</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>name/cos:GetObject</code></td>
<td>GET Object</td>
</tr>
<tr>
<td><code>name/cos:PutObject</code></td>
<td>PUT Object</td>
</tr>
<tr>
<td><code>name/cos:HeadObject</code></td>
<td>HEAD Object</td>
</tr>
<tr>
<td><code>name/cos:DeleteObject</code></td>
<td>DELETE Object</td>
</tr>
</tbody>
</table>

The following example specifies an action that is allowed:
"action": [ "name/cos:GetObject", "name/cos:HeadObject" ]

**Specifying a Resource**

A resource element describes one or multiple action objects including COS buckets and objects. All the resources can be described in the following six-piece format.

```
qcs:project_id:service_type:region:account:resource
```

Parameter description:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>qcs</td>
<td>The abbreviation of qcloud service, which refers to Tencent Cloud services.</td>
<td>Yes</td>
</tr>
<tr>
<td>project_id</td>
<td>Describes the project information, which is only used to enable compatibility with legacy CAM logic.</td>
<td>No</td>
</tr>
<tr>
<td>service_type</td>
<td>Describes the abbreviation of the product such as COS.</td>
<td>Yes</td>
</tr>
<tr>
<td>region</td>
<td>Describes the region information. For more information, see <a href="#">Regions &amp; Endpoints</a> supported by Tencent Cloud COS.</td>
<td>Yes</td>
</tr>
<tr>
<td>account</td>
<td>Describes the root account information of the resource owner. &quot;uin&quot; and &quot;uid&quot; can be used to describe a resource owner. The former is QQ number of the root account in the format of <code>uin/${OwnerUin}</code>, such as <code>uin/100000000001</code>. The latter is APPID of the root account in the format of <code>uid/${appid}</code>, such as <code>uid/1250000000</code>. For now, COS resource owner is always described using uid, i.e., the APPID of the root account.</td>
<td>Yes</td>
</tr>
<tr>
<td>resource</td>
<td>Describes the detailed resource information. In COS, a resource is described using the bucket XML API access domain name.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

The following example specifies the bucket examplebucket-1250000000.

```
"resource": ["qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/*"]
```

The following example specifies all objects in the `/folder/` folder in the bucket examplebucket-1250000000.

```
"resource": ["qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/folder/*"]
```

The following example specifies the `/folder/exampleobject` object in the bucket examplebucket-1250000000.

```
"resource": ["qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000/folder/exampleobject"]
```

**Specifying a Condition**

The access policy language allows you to specify conditions when granting permissions, such as limiting the access source of a user or the authorization time. The list below contains supported conditional operators as well as general condition keys and examples.

<table>
<thead>
<tr>
<th>Conditional Operator</th>
<th>Meaning</th>
<th>Condition Name</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ip_equal</td>
<td>IP is equal to</td>
<td>qcs:ip</td>
<td>{&quot;ip_equal&quot;:{&quot;qcs:ip &quot;:&quot;10.121.2.0/24&quot;}}</td>
</tr>
<tr>
<td>Ip_not_equal</td>
<td>IP is not equal to</td>
<td>qcs:ip</td>
<td>{&quot;ip_not_equal&quot;:{&quot;qcs:ip&quot;:[&quot;10.121.1.0/24&quot;,&quot;10.121.2.0/24&quot;]}}</td>
</tr>
</tbody>
</table>
### Conditional Operator

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
<th>Condition Name</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>date_not_equal</code></td>
<td>Time is not equal to</td>
<td><code>qcs:current_time</code></td>
<td><code>{&quot;date_not_equal&quot;:{&quot;qcs:current_time&quot;:&quot;2016-06-01T00:01:00Z&quot;}}</code></td>
</tr>
<tr>
<td><code>date_greater_than</code></td>
<td>Time is greater than</td>
<td><code>qcs:current_time</code></td>
<td><code>{&quot;date_greater_than&quot;:{&quot;qcs:current_time&quot;:&quot;2016-06-01T00:01:00Z&quot;}}</code></td>
</tr>
<tr>
<td><code>date_greater_than_equal</code></td>
<td>Time is greater than or equal to</td>
<td><code>qcs:current_time</code></td>
<td><code>{&quot;date_greater_than_equal&quot;:{&quot;qcs:current_time&quot;:&quot;2016-06-01T00:01:00Z&quot;}}</code></td>
</tr>
<tr>
<td><code>date_less_than</code></td>
<td>Time is less than</td>
<td><code>qcs:current_time</code></td>
<td><code>{&quot;date_less_than&quot;:{&quot;qcs:current_time&quot;:&quot;2016-06-01T00:01:00Z&quot;}}</code></td>
</tr>
<tr>
<td><code>date_less_than_equal</code></td>
<td>Time is less than or equal to</td>
<td><code>qcs:current_time</code></td>
<td><code>{&quot;date_less_than_equal&quot;:{&quot;qcs:current_time&quot;:&quot;2016-06-01T00:01:00Z&quot;}}</code></td>
</tr>
</tbody>
</table>

The following example specifies that the access IP is within the IP range of 10.121.2.0/24.

```
"ip_equal":{"qcs:ip ":"10.121.2.0/24"
```

The following example specifies that the access IPs are 101.226.***.185 and 101.226.***.186.

```
"ip_equal": {
  "qcs:ip": [ 
    "101.226.***.185",
    "101.226.***.186"
  ]
}
```

### Examples

If, when the access source IPs are 101.226.***.185/101.226.***.186, the root account allows anonymous users to perform GET (download) and HEAD actions on the objects in the bucket examplebucket-125000000 in South China, no authentication is required. For more information, see [Cases of Permission Setting](#).

}]
"effect": "allow",
"resource": [
"qcs::cos:ap-guangzhou:uid/1250000000:examplebucket-1250000000.ap-guangzhou.myqcloud.com/*"
]
]}
]
Examples of Bucket Policies

Overview

This document describes examples of bucket policies used to limit subnets, principals and VPC ID. You can use these bucket policies to control specific access to the bucket and its data. For more information, see Access Policy Language Overview.

- When configuring a bucket policy in the COS Console, you need to grant users appropriate permissions to the bucket, for example, getting the bucket location, and listing the bucket permissions.
- The bucket policy size is limited to 20 KB.

Example 1: Limit the IP range in the subnet to 10.1.1.0/24 and vpcid to aqp5jrc1.

Syntax example:

```json
{
  "Statement": [
    {
      "Action": ["name/cos:*"],
      "Condition": {
        "ip_equal": {
          "qcs:ip": ["10.1.1.0/24"]
        },
        "string_equal": {
          "vpc:requester_vpc": ["vpc-aqp5jrc1"]
        }
      },
      "Effect": "deny",
      "Principal": {
        "qcs": ["qcs::cam::anyone:anyone"]
      },
      "Resource": ["qcs::cos:ap-guangzhou:uid/1251668577:jimmyyantest-1251668577/*"]
    }
  ],
  "version": "2.0"
}
```

Example 2: Limit vpcid to aqp5jrc1 and specify the principal and bucket.

Syntax example:

```json
{
  "Statement": [
    {
      "Action": ["name/cos:*"],
      "Condition": {
        "ip_equal": {
          "qcs:ip": ["10.1.1.0/24"]
        },
        "string_equal": {
          "vpc:requester_vpc": ["vpc-aqp5jrc1"]
        }
      },
      "Effect": "deny",
      "Principal": {
        "qcs": ["qcs::cam::anyone:anyone"]
      },
      "Resource": ["qcs::cos:ap-guangzhou:uid/1251668577:jimmyyantest-1251668577/*"]
    }
  ],
  "version": "2.0"
}
```
"Action": [ "name/cos:*" ],
"Condition": { "string_equal": { "vpc:requester_vpc": [ "vpc-aqp5jrc1" ] } },
"Effect": "allow",
"Principal": { "qcs": [ "qcs::cam::uin/3280754170:uin/100005212150" ] },
"Resource": [ "qcs::cos:ap-beijing:uid/1252921383:x3cmplay-1252921383/*" ] },
"version": "2.0" }
Access strategy evaluation process

Access COS Bucket and Bucket resources need to be authorized before Access. In Tencent Cloud's Permission system, the main account to which the resources belong owns all the management Permission of the resources in Bucket and Bucket by default. CAM users (other main accounts, collaborators, sub-accounts) and other types of users, such as anonymous users, need to be authorized by the main account to carry out Access.

The Access policy in the account includes different policy types, such as user group policy, user policy, Bucket Access control list (ACL) and Bucket policy (Policy). In the evaluation of Access's strategy, there are the following key factors:

1. User Identity verification: when a user Access has resources on COS, there are two situations.
   - If you carry the request signature, COS will parse the user's account information from the request signature, and then request repost to Identity verification Access's management CAM.
   - If you do not carry a signature request, it will be considered as an anonymous user to enter the next link of Verification.
2. The category of Access strategy: Access strategy includes many types of strategies, such as user group, user, Bucket and so on. The category of Access strategy determines the order of Access strategy.
3. Policy context information: when processing a request for user resource Access, it will make a joint judgment based on the details of Permission recorded in user group policy, user policy, Bucket policy and other policies, and finally decide whether to pass the request or not.

Access strategy evaluation process

When Tencent Cloud COS receives a request, it will first confirm the identity of the requester and verify whether the requester has the relevant Permission. The verification process includes checking the user policy, Bucket Access policy and resource-based Access control list to authenticate the request.

When Tencent Cloud COS receives the request, Identity verification will be used first. According to the result of Identity verification, the identity of the requestor will be classified. Different identity categories may have different performance:

1. Verified Tencent Cloud main account: the main account has all the operations on the resources to which it belongs, Permission. As for the resources that do not belong to it, it is necessary to evaluate the resources Permission, and if the authentication is passed, the resources of Access will be licensed.
2. Authenticated CAM users (sub-accounts or collaborators): evaluate user policy-CAM users must have the authorization of their parent primary account before they are allowed to initiate the relevant Access. If the CAM user needs the resources to which Access's other main account belongs, you need to evaluate the resource Permission of the main account to which the CAM user belongs. If the authentication is passed, Access resources will be granted.
3. Anonymous users without identity: evaluation resource Permission-Permission who evaluates the Bucket Access strategy or Bucket or the object's Access control list. If the authentication is passed, the Access resource is granted.
4. Requesters other than the above categories of users: reject their Access.

Access's strategy evaluation basis

In the Tencent Cloud Permission system, in the Access strategy evaluation process, Permission is evaluated according to the policy context information, and there are the following basic principles:

1. By default, all requests are implicitly rejected by Access Permission, who is entitled to all the resources under the (deny); main account by default.
2. This default value is overridden if Explicit allows it in the user group policy, user policy, Bucket policy, or Bucket / object Access control list.
3. Explicit's refusal in any strategy will override any permission.
4. The effective Permission scope is the combination of identity-based policies (user group policy, user policy) and resource-based policies (Bucket policy or Bucket / object Access control list).

- **Explicit refuses:** there is a clear Deny policy for specific users in user policy, user group policy and Bucket Policy. If the main account explicitly configures Deny sub-user UIN 1000000011 in the user policy `GET Object` Operation, the sub-user cannot download the object resources under the main account.
- **Explicit allows:** pass in user policy, user group policy, Bucket Policy, Bucket ACL `grant-` Explicitly specify that a specific user has a clear Allow policy for a specific user.
- **Reject everyone:** clearly specified in Bucket Policy `Deny anyone` After everyone is rejected, any request without signature will be rejected, and the request with signature will be authenticated with identity-based policy.
- **Allow everyone:** explicitly specified in Bucket Policy `Allow anyone` Or explicitly specified in Bucket ACL `public-`.
- **Effective Permission's scope** is the combination of identity-based policy and resource-based policy: in a complete authentication, COS will first resolve the user's identity and verify that he has the resources of Permission and Access according to the user's identity; at the same time, according to the resource-based policy, the user will be treated as an anonymous user for Permission verification. Access will be allowed if one of the two verifications is successful.

Access's policy evaluation is based on the following figure. First, whether the user is an anonymous user is evaluated based on whether the signature is carried in the request. If the user is an anonymous user, it will evaluate whether there is a policy of rejecting or allowing everyone in the policy, and then decide whether to allow Access or reject Access. If the user is Valid's CAM user or the main account with resources, it will evaluate whether there is a policy in which Explicit rejects or Explicit allows or allows everyone, and then decides whether to allow Access or reject Access.

**Policy context information**

Policy context information refers to the details of Permission recorded in the policy. Under, the user needs to explicitly specify the following information in the policy:

- **The principal (principal):** must specify which sub-users you need to grant (Enter user ID), collaborator (Enter user ID), anonymous user or user group Permission. If you use a temporary key for Access, you do not need to specify this item.
- **The statement (statement):** specifies Enter's corresponding parameters in the following parameters.
- **Effective (effect):** must clearly state whether the policy is "allowed" or "rejected by Explicit", including both allow and deny.
- **The action (action):** must explicitly declare the actions that are allowed or denied by the policy. The operation can be API (described with the name prefix) or a feature set (a specific set of API, described with the permid prefix).
- **The resource (resource):** must explicitly declare the specific resources authorized by the policy. Resources are described in six paragraphs. You can specify the scope of the resource to the specified file, such as exampleobject.jpg, or the specified Directory, such as examplePrefix/*. Unless the business needs, please do not arbitrarily grant the user Access all the resources of Permission, that is, wildcards *.
- **Conditional (condition):** describes the constraints under which the policy takes effect. Conditions include operators, action keys, and operation values. Condition values can include information such as time, IP address, and so on.

To write a policy, you need to write it according to a certain policy syntax, see Access Policy Language Overview Write according to the needs of the business. For examples of user policy and Bucket policy, please see User policy syntax structure and Examples of Bucket Policies.

**Access's example of Strategy Evaluation**

Suppose the main account UIN 10000000001 is the sub-account UIN 10000000011. Associate has a user default policy that allows the sub-account to read only the resources under the main account. The details of this user policy are as follows. This user policy
allows the sub-account to execute all List, Get, Head Operation and OptionsObject Operation.

```json
{
    "version": "2.0",
    "statement": [
        {
            "action": [
                "cos:List*",
                "cos:Get*",
                "cos:Head*",
                "cos:OptionsObject"
            ],
            "resource": "*",
            "effect": "allow"
        }
    ]
}
```

At the same time, the main account is read and written by a privately owned Bucket examplebucket-1250000000. The following Bucket strategy has been added:

```json
{
    "Statement": [
        {
            "Principal": {
                "qcs": [
                    "qcs::cam::anyone:anyone"
                ],
            },
            "Effect": "Deny",
            "Action": [
                "name/cos:GetObject"
            ],
            "Resource": [
                "qcs::cos:ap-guangzhou:uid/1000000001:examplebucket-1250000000/*"
            ]
        }
    ],
    "version": "2.0"
}
```

This Bucket strategy explicitly refused all users to execute the download object (GetObject). Therefore, in the Access strategy evaluation process:

1. If the sub-account carries a signature parameter request GetObject The corresponding user policy will be matched according to the user identity represented by the request, and Access policy evaluation and verification will pass.
2. If the sub-account does not carry a signature parameter request GetObject Will be judged as an anonymous request by the system and will be strategically implemented by Bucket Reject Access.
ACL-based Access Control

ACLS Overview

Concept

An ACL is described in the XML language. It is a list of specified grantees and permissions granted, which is associated with resources. Each bucket and object has an associated ACL to grant basic read and write permissions to anonymous users or other Tencent Clouds root accounts.

Here are some limits on the use of ACLs associated with resources.

- The resource owner always has FULL_CONTROL permission on the resource, which cannot be revoked or modified.
- An anonymous user cannot be the resource owner. In this case, the object owner is the bucket owner (Tencent Cloud root account).
- Permissions can only be granted to Tencent Cloud CAM root accounts or anonymous users, rather than sub-users or user groups.
- No conditions should be imposed on the permissions.
- "Deny" permission is not supported.
- A resource can have up to 100 ACL policies.

ACL Elements

Identity (Grantee)

A supported grantee can be a CAM root account or a preset CAM user group.

- When you grant access permissions to another Tencent Cloud root account, this root account can grant access permissions to its sub-users, user groups or roles.
- In COS, it is strongly recommended not to grant WRITE, WRITE_ACP or FULL_CONTROL permission to any anonymous user or CAM user group. Otherwise, the user group can upload, download or delete your resources, which will bring security risks to your account, such as data loss and fee deduction.

Grantees supported in bucket or object ACLs include:

- Cross-account: Use the root account ID to get the "Account ID" in Account Info, e.g. 388626565.
- Preset user group: Tag a preset user group using a URI tag. The following user groups are supported:
  - Anonymous user group - http://cam.qcloud.com/groups/global/AllUsers, which indicates that anyone can access resources without authorization, regardless of whether the request is signed or unsigned.
  - Certified User Group - http://cam.qcloud.com/groups/global/AuthenticatedUsers, which indicates that users that have registered a Tencent Cloud CAM account can access resources.

Action permissions

Tencent Cloud COS supports a range of action sets on resource ACLs, which include different actions on bucket ACLs and object ACLs respectively.

Actions on buckets
Here is a list of actions supported in the bucket ACLs:

<table>
<thead>
<tr>
<th>Action Set</th>
<th>Description</th>
<th>Allowed Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>READ</td>
<td>Lists objects</td>
<td>GetBucket, HeadBucket, GetBucketObjectVersions, ListMultipartUploads</td>
</tr>
<tr>
<td>WRITE</td>
<td>Uploads, overwrites and deletes objects</td>
<td>PutObject, PutObjectCopy, PostObject, InitiateMultipartUpload, UploadPart, UploadPartCopy, CompleteMultipartUpload, DeleteObject</td>
</tr>
<tr>
<td>READ_ACP</td>
<td>Reads bucket ACLs</td>
<td>GetBucketAcl</td>
</tr>
<tr>
<td>WRITE_ACP</td>
<td>Writes to bucket ACLs</td>
<td>PutBucketAcl</td>
</tr>
<tr>
<td>FULL_CONTROL</td>
<td>A collection of the above sets</td>
<td>A collection of the above actions</td>
</tr>
</tbody>
</table>

Please proceed with caution when you grant WRITE, WRITE_ACP or FULL_CONTROL permission on buckets. Granting the WRITE permission will allow the grantee to overwrite or deleted any existing object.

Actions on objects

Here is a list of actions supported in the object ACLs:

<table>
<thead>
<tr>
<th>Action Set</th>
<th>Description</th>
<th>Allowed Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>READ</td>
<td>Reads objects</td>
<td>GetObject, GetObjectVersion, HeadObject</td>
</tr>
<tr>
<td>READ_ACP</td>
<td>Reads object ACLs</td>
<td>GetObjectAcl, GetObjectVersionAcl</td>
</tr>
<tr>
<td>WRITE_ACP</td>
<td>Writes to object ACLs</td>
<td>PutObjectAcl, PutObjectVersionAcl</td>
</tr>
<tr>
<td>FULL_CONTROL</td>
<td>A collection of the above sets</td>
<td>A collection of the above actions</td>
</tr>
</tbody>
</table>

The WRITE operation set is not supported for objects.

Preset ACL

Tencent Cloud COS supports a group of preset ACLs for authorization, making it much easier to describe simple permissions. When using a preset ACL for description, you need to put the `x-cos-acl` header in the PUT Bucket/Object or PUT Bucket/Object ACL and describe the required permission. If an XML description is also carried in the request body, the description in the header is used and the XML description in the request body is ignored.

Preset ACLs for buckets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>The creator (root account) has the FULL_CONTROL permission, while others have no permissions. (Default)</td>
</tr>
<tr>
<td>public-read</td>
<td>The creator has the FULL_CONTROL permission, and the anonymous user group has the READ permission.</td>
</tr>
<tr>
<td>public-read-write</td>
<td>Both the creator and the anonymous user group have the FULL_CONTROL permission, which is generally not recommended.</td>
</tr>
<tr>
<td>authenticated-read</td>
<td>The creator has the FULL_CONTROL permission, and the certified user group has the READ permission.</td>
</tr>
</tbody>
</table>

Preset ACLs for objects
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>An empty description. The request follows the bucket or sub-user policy explicitly allowed. No declaration means deny.</td>
</tr>
<tr>
<td>private</td>
<td>The creator (root account) has the FULL_CONTROL permission, while others have no permissions.</td>
</tr>
<tr>
<td>public-read</td>
<td>The creator has the FULL_CONTROL permission, and the anonymous user group has the READ permission.</td>
</tr>
<tr>
<td>authenticated-read</td>
<td>The creator has the FULL_CONTROL permission, and the certified user group has the READ permission.</td>
</tr>
<tr>
<td>bucket-owner-read</td>
<td>The creator has the FULL_CONTROL permission, and the bucket owner has the READ permission.</td>
</tr>
<tr>
<td>bucket-owner-full-control</td>
<td>Both the creator and the bucket owner have the FULL_CONTROL permission.</td>
</tr>
</tbody>
</table>

The public-read-write permission is not supported for objects.

**Example**

**Bucket ACL**

When you create a bucket, COS creates a default ACL to grant the resource owner the full control over the resource (FULL_CONTROL), as shown in the following example:

```xml
<AccessControlPolicy>
  <Owner>
    <ID>Owner-Cononical-CAM-User-Id</ID>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>Owner-Cononical-CAM-User-Id</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

**Object ACL**

When you create an object, COS does not create an ACL by default, and the object owner is the bucket owner. The object's permissions inherited from its bucket are the same with the access permissions of the bucket. Since the object does not have a default ACL, it will follow the definition of visitors and behaviors in the bucket policy to determine if the request is granted. For more information, see [Access Policy Language Overview](#).

If you need to grant additional access permissions to the object, you can add more ACLs to describe the access permissions of the object. For example, grant an anonymous user the permission to read a single object, as shown below:

```xml
<AccessControlPolicy>
  <Owner>
    <ID>Owner-Cononical-CAM-User-Id</ID>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>anonymous</ID>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```
<AccessControlPolicy>
  <Grant>
    <Grantee>
      <ID>Owner-Cononical-CAM-User-Id</ID>
    </Grantee>
    <Permission>FULL_CONTROL</Permission>
  </Grant>
  <Grant>
    <Grantee>
      <URI>http://cam.qcloud.com/groups/global/AllUsers</URI>
    </Grantee>
    <Permission>READ</Permission>
  </Grant>
</AccessControlList>
</AccessControlPolicy>
Tag-based Access Control
Manage project resources based on tags

Overview

This document describes how to use tags and tag authentication to manage project resources in the tag system. It is applicable to the scenario where a user used projects in the legacy console and granted sub-accounts access through project authentication.

Project management is centralized management of resources in the project dimension. You can add Tencent Cloud product resources that support project management to a project. Then, select **Policy > Create Custom Policy > Create by Product Feature or Project Permission** in the **CAM Console** to generate a project policy. You can associate the project policy with project-related users or user groups, so that they can get the permission to manipulate project resources.

The legacy COS Console enables permission management operations based on projects. However, a project policy grants the full access to all resources under all products included in the project. It **cannot meet the needs for multi-dimensional tagging and categorization, nor can it manage permissions in a refined manner**. In the new COS Console, COS only supports tag-based permission management for project resources.

COS uses the tag service to be compatible with the legacy project feature. In the tag service system, a project is a special tag with the tag key being **project**. You can still create a project and then create a bucket under it in the project console. COS will automatically double-write the project affiliation of a bucket during bucket creation to the tag service, so that the affiliation can be displayed in the console.

- If you need to manage buckets in a categorized manner, you are recommended to directly use tags instead of projects to implement tasks such as permission control and bill splitting. For more information on how to add tags in the console, please see **Setting Bucket Tags**.
- For more information on the tag service, please see **Tag**.

Granting a Sub-account Access to a Project

You can grant a sub-account access to a project by following the steps below:

1. Log in to the **project console**, create a project, name it, and submit it. Then, create a resource such as a bucket or CVM instance under it. If you already have a project with storage or computing resources, you can skip this step.
2. After you create a project and bind a resource to it, enter the **Policy Management** page and click **Create Custom Policy > Authorize by Tag**. You can select the tag authentication method. Then, select the corresponding project tag and grant the sub-
account access to all resources under it.

3. The default policy is to grant the sub-account access to all resources under the project tag. If you want the sub-account to perform only specified operations on certain resources under the tag, you can change the `action` (for specifying operations) and `resource` (for specifying resources) in the policy syntax as instructed in Syntax Structure and click Complete.
4. If you want the sub-account to be able to create buckets, you also need to grant it the PUT Bucket permission. On the Policy Management page, click Create Custom Policy > Create by Policy Generator and grant the sub-account the corresponding permission.
Using CDN to Accelerate Access

CDN Acceleration Overview

Accelerating COS with CDN allows you to mass download and deliver the content in a bucket, especially in the use cases where the same content is repeatedly downloaded. With the origin-pull authentication feature, the delivery of content in a Public-Read bucket can be accelerated using CDN, and with the CDN authentication feature, the content can only be downloaded by legitimate users, thus avoiding data security problem and high traffic cost caused by unrestricted download access.

Content Delivery Network (CDN)

Definition of CDN

Content Delivery Network (CDN) is a layer of network architecture built on the Internet that is composed of globally distributed servers which work together to provide faster delivery of Internet content. These high-performance cache nodes store your Internet content based on a certain caching policy. When your user makes a request for your Internet content, the request will be routed to the server closest to the user. Then the server responds to the request directly, greatly reducing the user's access delay and improving availability.

Caching and origin-pull will occur on a CDN. When a user accesses a URL, if the requested content is not cached on the edge server to which the access request is directed, or the cached content has expired, the request will be returned to the origin server to get the content.

Use cases

- Scenarios with high requirements for response delay and download speed.
- Scenarios where cross-region, cross-border or cross-continent data transfer reaching GB or TB-level is required.
- Scenarios where the same content needs to be downloaded frequently.

Security options

- Origin-pull authentication: If the data requested by a user is not cached in the edge server, CDN fetches the data from the origin server. If COS is used as the origin server and origin-pull authentication is enabled, the CDN edge server accesses the COS origin server using a special service identity to acquire and cache the data in the private bucket.
- CDN service authorization: A CDN edge server can access the COS origin server with a special service identity by adding a CDN service authorization. Origin-pull authentication can only be enabled after the CDN service authorization is added.
- CDN authentication: When a user attempts to acquire cached data by accessing an edge server, the edge server verifies the authentication field in the accessed URL based on the authentication configuration rules to prevent unauthorized access and realize hotlink protection, thus improving the security and reliability of the data cached in the edge server.

Access Nodes of COS

Definition of access node

An access node is an access domain name that is assigned to a bucket based on the region and name of the bucket when it is created. The domain name can be used to access the data in the bucket.

If the static website feature is enabled, you can get an access node for a static website to present the specially configured response content that is different from that of the default node.

Access nodes
XML node: After a bucket is created, COS assigns an XML access node to the bucket, such as `<bucketname>-<APPID>.cos.<region>.myqcloud.com`. It is suitable for RESTful API calls. With an XML access node, you can configure a bucket or upload/download objects as described in API documentation.

Static website node: You can enable the static website hosting feature in the basic configuration interface of bucket in the console. After the feature is enabled, an access node in the form of `<bucketname>-<APPID>.cos-website.<region>.myqcloud.com` is provided. Static websites support special index pages (IndexPage), error pages (ErrorPage), and redirects, and only allow the download of objects. You can obtain content through static website nodes.

Access permissions

- Public Read: When a bucket is set to "Public Read", anyone can access it using the bucket's access domain name. If you use a public-read bucket as the origin server, you can enable CDN acceleration directly without using CDN authentication and origin-pull authentication.
- Private Read: When a bucket is set to "Private Read", you can manage accessing users and CDN service authorizations by creating access policies. When you use a private-read bucket as the origin server, if origin-pull authentication is enabled but CDN authentication is not, unauthorized users can access the bucket via CDN. Therefore, it is highly recommended to enable both origin-pull authentication and CDN authentication for private-read buckets to ensure the data security.

Accelerating Access to COS Using CDN

You can accelerate access to COS by managing the following two domain names:

- Default accelerated domain name: COS provides a default CDN accelerated domain name (such as `<bucketname>-<APPID>.file.mycloud.com`). You can enable or disable it at your option.
- Custom domain name: You can use a custom domain name which has gone through ICP filing, with a COS bucket as the origin server. This allows you to accelerate the access to the objects in the bucket with your custom domain name.

**Notes:**
Default accelerated domain name and custom domain name can be collectively referred to as CDN accelerated domain names.

Public-read buckets

When a bucket is set to allow public access, and the CDN origin server is set to the COS access node, the CDN edge servers can acquire and cache the object data in the bucket without you enabling origin-pull authentication.

You can provide **limited** protection for the data in the bucket by enabling Authentication Configuration in the CDN Console. This is because that regardless of whether this feature is enabled in the CDN, the users who know the bucket access domain name can access all objects in the bucket. Whether the access to public-read buckets is possible via different domain names when CDN authentication is enabled or disabled is as follows:

<table>
<thead>
<tr>
<th>CDN Authentication</th>
<th>CDN Accelerated Domain Name</th>
<th>COS Domain Name</th>
<th>Scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disabled (default)</td>
<td>Yes</td>
<td>Yes</td>
<td>Public access to the entire website via CDN or origin server is allowed.</td>
</tr>
<tr>
<td>Enabled</td>
<td>URL authentication is required</td>
<td>Yes</td>
<td>Hotlink protection is enabled for access via CDN access, but not for access via origin server (not recommended)</td>
</tr>
</tbody>
</table>

Private-read buckets
When a bucket defaults to Private Read, and the CDN origin server is set to the COS access node, the CDN edge nodes are unable to get and cache any data. Therefore, you need to add the CDN service identity to the Bucket Policy and authorize the identity to perform the following operations:

- GET Object - Acquire objects.
- HEAD Object - Acquire the metadata of objects.
- OPTIONS Object - Initiate a preflight request.

You can complete quick authorization in both the CDN Console and the COS Console by simply clicking Add CDN Service Authorization. Then you need to enable Origin-Pull Authentication. After that, a CDN edge server can access the data in the COS with its service identity.

Notes:
1. If the bucket is set to Private Read, you must add an authorization and enable origin-pull authentication, otherwise COS will deny the access to it.
2. A CDN edge server will generate a unique service account for each root account. Therefore, the account authorization is only valid for the root account to which the accelerated domain name belongs. Cross-account binding of an accelerated domain name will cause the access via the domain name to be denied.

After the CDN service authorization is added and origin-pull authentication is enabled, the CDN edge nodes are able to directly get and cache the data. Therefore, it is highly recommended to enable Authentication Configuration to protect the private data in a bucket. Whether the access to private-read buckets is possible via different domain names when CDN authentication is enabled or disabled is as follows:

<table>
<thead>
<tr>
<th>CDN Authentication</th>
<th>CDN Accelerated Domain Name</th>
<th>COS Domain Name</th>
<th>Scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disabled (default)</td>
<td>Yes</td>
<td>COS authentication is required</td>
<td>Direct access to CDN domain names is allowed to protect the data on origin server.</td>
</tr>
<tr>
<td>Enabled</td>
<td>URL authentication is required</td>
<td>COS authentication is required</td>
<td>Full stack strict SSL secured connection. Hotlink protection for CDN authentication is supported.</td>
</tr>
</tbody>
</table>
### CDN Acceleration Configuration

**Last updated : 2019-09-30 18:09:01**

### Applicable Scenarios

- Scenarios that requires low latency and high download speed.
- Scenarios where gigabytes to terabytes of data need to be transmitted across regions, countries, and continents.
- Scenarios where the same content needs to be downloaded frequently and repeatedly.

### Notes

For information on domain name definition, CDN origin-pull authentication, and CDN authentication configuration, see [CDN Acceleration Overview](#).

CDN origin-pull authentication and CDN authentication configuration affect how CDN accelerated domain names and COS domain names access origin server buckets. Please find the details in the table below.

<table>
<thead>
<tr>
<th>Bucket Access Permission</th>
<th>CDN Origin-pull Authentication</th>
<th>CDN Authentication Configuration</th>
<th>CDN Accelerated Domain Name</th>
<th>COS Domain Name</th>
<th>Scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public read</td>
<td>Disabled</td>
<td>Disabled</td>
<td>Accessible</td>
<td>Accessible</td>
<td>Site-wide public access</td>
</tr>
<tr>
<td>Public read</td>
<td>Enabled</td>
<td>Disabled</td>
<td>Accessible</td>
<td>Accessible</td>
<td>Not recommended</td>
</tr>
<tr>
<td>Public read</td>
<td>Disabled</td>
<td>Enabled</td>
<td>URL authentication required</td>
<td>Accessible</td>
<td>Not recommended</td>
</tr>
<tr>
<td>Public read</td>
<td>Enabled</td>
<td>Enabled</td>
<td>URL authentication required</td>
<td>Accessible</td>
<td>Not recommended</td>
</tr>
<tr>
<td>Private read + CDN service authorization</td>
<td>Enabled</td>
<td>Enabled</td>
<td>URL authentication required</td>
<td>COS authentication required</td>
<td>Full-linkage protection</td>
</tr>
<tr>
<td>Private read + CDN service authorization</td>
<td>Disabled</td>
<td>Enabled</td>
<td>Inaccessible</td>
<td>COS authentication required</td>
<td>Not recommended</td>
</tr>
<tr>
<td>Private read + CDN service authorization</td>
<td>Enabled</td>
<td>Disabled</td>
<td>Accessible</td>
<td>COS authentication required</td>
<td>Origin server protection</td>
</tr>
<tr>
<td>Private read + CDN service authorization</td>
<td>Disabled</td>
<td>Disabled</td>
<td>Inaccessible</td>
<td>COS authentication required</td>
<td>Not recommended</td>
</tr>
<tr>
<td>Private read</td>
<td>Disabled</td>
<td>Enabled or disabled</td>
<td>Inaccessible</td>
<td>COS authentication required</td>
<td>Unable to use CDN</td>
</tr>
</tbody>
</table>
Configuring CDN Acceleration for a Default Accelerated Domain Name

A default accelerated domain name is a CDN accelerated domain name COS automatically assigns to a bucket. The format is <BucketName-APPID>.file.mycloud.com. You can enable or disable it as needed.

Enabling the Feature

1. Select a bucket
Log in to the COS Console, click Bucket List in the left sidebar, and click the bucket to be accelerated to enter the bucket.

2. Enter the configuration page

If you have never used Tencent Cloud CDN service before, please log into the CDN Console and activate the CDN service first.

Click Domain Name Management on the left of the page. The default status is off. Click Edit and toggle it on.

Origin Server Type: The Default origin server is selected by default. If a static website is enabled for the bucket serving as the origin server and you want to accelerate the static website, you can set the origin server type to Static website origin server. For more information, see CDN Acceleration Overview.

3. Add a CDN service authorization (optional)
Adding a CDN service authorization is to grant a CDN edge server the service identity which allows it to perform operations on a bucket. Please find the details below.

- Public read bucket: the CDN edge server can access the bucket without any authorization, so there is no need to add a CDN service authorization.
Private read bucket: the CDN edge server needs a special service identity to access the bucket. Click **Add a CDN Service Authorization** to grant the CDN edge server the service identity, select **I agree to the authorization above**, and click **OK**.

After the authorization is added, the CDN edge server can perform GET Object, Head Object, and Options Object on the bucket.

![Authorization Form](image)

After the authorization is granted, it will be automatically written into the bucket access policy (see below for an example). Then the CDN edge server does not need to do anything else while forwarding traffic to the origin.

```json
{
   "Statement": [
      {
         "Action": [
            "name/cos:GetObject",
            "name/cos:HeadObject",
            "name/cos:OptionsObject"
         ],
         "Effect": "allow",
         "Principal": {
            "qcs": [
               "qcs::cam::uin/100000000001:service/cdn"
            ],
            "qcs::cos:ap-chengdu:uid/1250000000:examplebucket-1250000000/*"
         ],
         "Resource": [
            "qcs::cos::ap-chengdu:uid/1250000000:examplebucket-1250000000/*"
         ]
      }
   ],
   "version": "2.0"
}
```

4. Enable origin-pull authentication (optional)

You need to add the CDN service authorization before you can enable origin-pull authentication.

Origin-pull authentication is used to verify the service identity of the CDN edge server so as to prevent unauthorized access. Please find the details below.
Public read bucket: the CDN edge server can access the bucket without any authorization, so there is no need to enable origin-pull authentication.

Private read bucket: the CDN edge server needs to be verified by origin-pull authentication. Only verified edge server can access the objects in the bucket. To enable the authentication, toggle on **Origin-pull Authentication** and click **Save**.

For private read buckets, after origin-pull authentication and CDN service authorization are enabled, the CDN edge server will no longer need to carry a signature when accessing the origin server, and the resources cached by CDN will be delivered over the public network. In such case, data security will be under threat. Therefore, it is highly recommended to enable CDN authentication.

### 5. Enable CDN acceleration

After clicking **Save**, you will see that the default accelerated domain name is being deployed (which is expected to be completed in about 5 minutes).

**Configuring Authentication**

After CDN acceleration is enabled for the default domain name, anyone can access the origin server through this domain name. If your data requires privacy, please be sure to enable the authentication to protect your data on the origin server.

After the default accelerated domain name and origin-pull authentication are enabled, a CDN authentication status prompt will appear on the default accelerated domain name management page. You can click **authentication configuration** in the prompt to
enter the CDN security configuration page for the corresponding domain name.

You can also go to this page from the CDN Console by clicking Domain Name Management > Management (for the corresponding domain name) > Security Configuration. For detailed configuration directions, see [Authentication Configuration].

**Disabling the Feature**

- On the default accelerated domain name management page, click Edit, toggle the status from on to off, and click Save. The change will be deployed in around 5 minutes. After that, the domain name will be displayed as Deactivated in the CDN Console.

- You can deactivate or delete the domain name in the CDN Console. For more information, see Domain Name Operations.

When you delete a domain name in CDN console, you are deleting a CDN acceleration record of the default accelerated domain name, not the domain name itself. You can activate it again in the COS console.

**Configuring CDN Acceleration for a Custom Accelerated Domain Name**

You can bind a custom domain name with a bucket in the COS Console. After that, you can enable CDN acceleration to speed up access to the bucket through the custom domain name. When binding a custom domain name with a bucket, you need to add a CNAME record to it at your domain name service provider.

Currently, you need to enable CDN service to use a custom domain name in COS.

1. For domain names connected to a CDN node in Mainland China, you need to complete ICP filing. You are not required to do so through Tencent Cloud though.
2. For domain names connected to a CDN node outside Mainland China, ICP filing is not required, but please note that your data and operations in Tencent Cloud still need to comply with local laws and regulations as well as Tencent Cloud Service Agreement.
Enabling the Feature

You can add a custom domain name and enable CDN acceleration in both COS Console and CDN Console. For more information on how to do so in the CDN Console, see Connect a Domain Name.

1. Select the bucket to be bound with a custom domain name
Log in to the COS Console, click Bucket List in the left sidebar, and click the bucket to be accelerated to enter the bucket.

2. Add a custom domain name
Click Domain Name Management to enter the domain name management page. Click Add a Domain Name in the custom domain name section to start the configuration.

   **Domain Name**: Enter the domain name you have purchased (such as www.example.com).
   **Origin Server Type**: The Default origin server is selected by default. If a static website is enabled for the selected bucket and you want to accelerate the static website, you can set the origin server type to Static website origin server.

3. Enable origin-pull authentication (optional)
Origin-pull authentication is used to verify the service identity of the CDN edge server so as to prevent unauthorized access. Please find the details below.

   - Public read bucket: the CDN edge server can access the bucket without any authorization, so there is no need to enable origin-pull authentication.
   - Private read bucket: the CDN edge server needs to be authenticated by origin-pull authentication. Only verified edge server can access the objects in the bucket. To enable the authentication, toggle on Origin-pull Authentication.

For private read buckets, after origin-pull authentication is enabled, the CDN edge server will no longer need to carry a signature when accessing the origin server, and the resources cached by CDN will be delivered over the public network. In such case, data security will be under threat. Therefore, it is highly recommended to enable CDN authentication.
4. Enable CDN acceleration

Click **Save** on the right. In about 5 minutes, the added custom domain name and CDN acceleration will be deployed.

**Configuring Authentication**

After CDN acceleration is enabled for the custom domain name, anyone can directly access the origin server through this domain name. If your data requires privacy, please be sure to enable the authentication configuration to protect your data on the origin server.

After the custom domain name is deployed, a CDN authentication configuration link will appear in the CDN Authentication column. Click **Settings** to directly enter the CDN Console for CDN authentication configuration. For detailed directions, see [Authentication Configuration].

**Resolving a Domain Name**

After the custom domain name is connected to CDN, you will be assigned a CNAME domain name (suffixed with `.cdn.dnsv1.com`). You will need to complete the CNAME configuration through your domain name service provider. For more information, see CNAME Configuration.

You will not be able to directly access the CNAME domain name.

**Disabling the Feature**

On the accelerated custom domain name management page, click **Edit** to change the domain name status from **Enabled** to **Disabled**, click **Save**, and wait for the deployment to be completed in around 5 minutes. After that, the status of the accelerated custom domain name will be **Deactivated** instead of **Activated** in the CDN Console.

You can only delete a custom domain name when its status is **disabled**. If the status is **enabled**, you will need to change it to **disabled** first.
You can deactivate or delete domain names in the CDN Console. For more information, see Domain Name Operations.
Configuring GCD

Overview

Tencent Cloud Global Content Delivery (GCD) distributes Tencent's high-performance cache nodes globally, which accelerate worldwide static content, content download and delivery, audio/video on-demand services. Tencent Cloud GCD is under open beta test and the quota is limited. Submit an application if you want to have a try. For more information, see How to apply for trial. The following describes how to configure GCD for CDN.

You need to apply for activating GCD before you can use this feature.

Use Cases

- The end users outside Mainland China need to be reached and accelerating access and reducing access delay are required.
- Cross-region, cross-border or cross-continent data transfer reaching GB or TB-level is required.
- The same content needs to be downloaded frequently.

Description

When your application for activating GCD is approved, go to Tencent Cloud's GCD Console to add COS's XML access domain name for acceleration.
You need to set the content to be accessed to Public in COS in order to bind and use the GCD accelerated domain name to access the content on COS.

Directions

Configure GCD by following the steps below: Configure Policy Permissions -> Configure GCD -> Configure CNAME.

Configure policy permissions

1. Log in to Tencent Cloud's COS Console to go to the management page. Click the Permission Management tab to add a public access policy.
2. Click **Add Policy**, and then set **User**, **Resource** and **Operation** as follows:
   - Effect: Allow
   - User: All Users
   - Resource: Entire Bucket
Operation: Read Access

Configure GCD

1. Obtain the COS domain name.
   In the COS Console, obtain the access domain name of the bucket for which you want to accelerate access.
   
   examplebucket-1250000000.cos.ap-guangzhou.myqcloud.com

   If you want to accelerate the delivery of content for a bucket for which the static website feature is enabled, and use the features of a static website, use a static website domain name, for example:

   examplebucket-1250000000.cos-website.ap-guangzhou.myqcloud.com

2. Bind a domain name to GCD
   Log in to Tencent Cloud's GCD Console, and then add a domain name by using COS as your self-owned origin. The configuration items are as follows:
   - Domain: Enter the access domain name you want to bind to GCD.
   - Service Type: The data in COS is mostly unstructured data. It is recommended to select Static Content.
   - Origin Type: Select External (self-owned origin).
Create a Distribution

- **Domain**: yourdomain.com
- **Service Type**: Static content
- **Project**: Default
- **Origin type**: External (Object storage (COS))
- **Origin IP/domain**: examplebucket-125.cos.ap-guangzhou.myqcloud.com
- **Host header**: examplebucket-125.cos.ap-guangzhou.myqcloud.com

Set other options as needed. For more information, see [Getting Started](#) in GCD documentation.

Configure CNAME

Go to your DNS service provider to add the CNAME record. For more information on how to configure CNAME, see [CNAME Configuration](#) in CDN documentation.

FAQs

For the question whether a domain name that has not gone through filing procedure with MIIT can be bound to GCD and other questions, see [FAQs](#) in COS documentation.
Single link speed limit

Last updated: 2020-04-22 12:12:17

Single-Link Speed Limit

COS supports traffic control for file uploads and downloads to guarantee normal network bandwidth for your other applications. You can add the `x-cos-traffic-limit` parameter in a request for `PutObject`, `PostObject`, `GetObject`, or `UploadPart` and set a speed limit value. Then, COS will control the network bandwidth for the request accordingly.

Instructions

- You can add the `x-cos-traffic-limit` parameter when requesting `Object`, `PostObject`, `GetObject`, or `UploadPart` to specify the speed limit value for the request. This parameter can be set in the header, request parameter, or form field (for upload through form).
- The value of the `x-cos-traffic-limit` parameter must be a number and the unit is bit/s by default.
- The speed limit value range is 819200–838860800, i.e, 100 KB/s–100 MB/s. If the range is exceeded, a 400 error will be returned.

API Use Case

The following is a sample request for the simple upload API with a speed limit of 1048576 bit/s (128 KB/s):

```
PUT /exampleobject HTTP/1.1
Host: examplebucket-1250000000.cos.ap-beijing.myqcloud.com
Content-Length: 13
Authorization: q-sign-algorithm=sha1&q-ak=AKID8A0fBVtYFrNm02oY1g1JQQF0c3J0****&q-sign-time=1561109068;1561116268&q-key-time=1561109068;1561116268&q-header-list=content-length;content-md5;content-type;date;host&q-url-param-list=&q-signature=998bfc8836fc285d09e455c14e3e623ed2****
x-cos-traffic-limit: 1048576
```
Batch Processing Overview

The batch operation feature of COS allows you to perform specified operations on the specified object list in the bucket. You can generate an inventory of objects through the inventory feature as the specified object list, or record the objects to be processed in a CSV file according to the format requirement of an inventory file. Then, COS will process the objects in batches based on the object inventory file.

For more information, please see Inventory Overview.

At present, the batch operation feature only supports the following operation:

- Replicating Objects in Batches
- Batch Resume archive object

You can use the batch operation feature in the COS Console. For more information, please see Batch Operation.

COS Select currently only supports public cloud regions in mainland China, not supported in other regions.

How It Works

To perform a batch operation, you need to create a batch operation job first, which contains all the information needed to perform the specified operation on the object list. You can use an inventory as an object list.

After you provide an object file and start the created batch operation job, the batch operation feature will execute the specified operation on the objects in the inventory sequentially. During job execution, you can monitor the execution status in the COS Console or choose to output a job report after the job is completed. The job report details the status of every operation in the job.

Object Inventory

An object inventory is a list of all the objects to be operated on. To create a batch operation job, you need to provide an object inventory first to tell COS what objects it should operate on. You need to put the object inventory file in the bucket and provide such information as the file name, ETag, and VersionID (if applicable). You can create an object inventory in the following two ways:

- **COS inventory feature**: This feature outputs an object inventory in CSV format. For more information, please see Inventory Overview. If your object inventory contains version ID information, COS will batch operate on the objects with the corresponding version ID.
- **CSV file configuration**: Every row in the CSV file must contain the bucket name and the name and version ID (optional) of an object for batch operation. If versioning has never been enabled, you can ignore the object version ID information. The CSV file can be configured as shown below:

```
examplebucket-appid, exampleobject, PZ
examplebucket-appid, exampleobject, PZ
examplebucket-appid, exampleobject, PZ
examplebucket-appid, exampleobject, PZ
examplebucket-appid, exampleobject, PZ
examplebucket-appid, exampleobject, PZ
```
If versioning is or was once enabled for your bucket, you must provide object versioning information in the object inventory. Before executing a job, COS will parse your inventory and determine the objects to be operated on based on the inventory.

An object inventory can contain all the objects in a bucket. However, it may take a longer time to operate on a large number of objects.

If version ID is not specified, COS will operate on the objects on the latest version by default. If you uploaded an object file with the same name as an object to be operated on before creating a job, COS will operate on the object on the latest version by default rather than the version when the object inventory was created. To avoid this issue, you can enable versioning and specify the version ID in the object inventory.

### Batch Operation Job

This section describes how to create a batch operation job and how the system will respond after creation.

When creating a batch operation job, you need to provide the following information:

- **Operation**
  You need to specify which operation is to be performed on the objects. The corresponding parameters can be configured for each operation, and COS will perform the operation on the objects in the inventory sequentially according to the configuration.

- **Object inventory**
  An object inventory is a list of all the objects to be operated. You can create an object inventory through the inventory feature. For more information, please see Inventory Overview. You can also record the objects to be processed in a CSV file according to the format requirement of an inventory file.

- **Priority**
  You can set priority to identify the precedence of the current batch operation job over other jobs. Job priority does not directly determine the order in which the jobs will be completed. If you want to control the order of multiple jobs, you need to check the job execution status on your own and start the next job after the current one is completed.

- **Rule permission**
  After creating a batch operation job, you need to make sure that your account has the corresponding IAM permission to perform the operation. For example, if you have created a batch operation job to execute `PUT Object-copy`, you should make sure that your account has the `Get Object` permission in the source bucket and the `PUT Object` permission in the destination bucket. In addition, for all batch operation jobs, you should have permission to read the object inventory and write into the job report. For more information on permission configuration, please see Access Policy Language Overview and Examples of Bucket Policies.

- **Job report**
  If you want to output a job report after a batch operation job is completed, you need to enter the corresponding parameters when creating the job, so that the system can correctly output the job report to the specified destination bucket. The required information includes the bucket to store the job report, job report format, and whether all job information should be included, while the file prefix of the job report is optional.

- **Job description (optional)**
  You can provide 256 bytes of job description for your created batch operation job to help track and monitor job execution. The detailed information of the job description will be displayed in your COS Console. You can sort or filter your jobs conveniently by job description. The same job description can be configured for similar jobs (e.g., syncing log data on a weekly basis) to manage them in a centralized manner.
Performing Batch Operation
Copying Objects in Batches

Last updated: 2020-01-15 15:20:01

The batch replication feature is used to replicate objects in the inventory, i.e., allowing you to batch replicate the specified objects from the source bucket to the destination bucket in the same region or in different regions. It supports customizing parameters for the PUT Object-copy operation, and the copy metadata and storage type are subject to the configuration information. For more information, please see PUT Object-copy.

Use Limits

- All objects to be replicated must be in the same bucket.
- Only one destination bucket can be configured for a batch replication job.
- You need to have permission to read objects from the source bucket and write objects into the destination bucket.
- The total size of the objects cannot exceed 5 GB.
- Verification via ETags and server-side encryption using custom keys are not supported.
- If the destination bucket does not have versioning enabled and contains an object file with the same name as a file to be replicated, COS will overwrite the object file.
Batch Resume archive object

Batch restoring archived objects can be used to restore objects of archived storage type in the inventory. This operation supports customizing parameters related to POST Object restore. These configuration information will affect the restoration and expiration time of the replica. For more information on POST Object restore, see [POST Object restore](https://intl.cloud.tencent.com/document/product/436/12633).

To create a job for batch restoring archived objects, you need to specify the following two parameters:

- **Restoration mode**: Standard or batch mode. For more information on restoration mode, see [Restoring Archived Objects](https://intl.cloud.tencent.com/document/product/436/12633).
- **Replica validity period**: When an object of archived type is restored, a temporary replica is created, which will expire and be deleted automatically after a specified time period. For more information on the validity period of a replica, see [Restoring Archive Objects](https://intl.cloud.tencent.com/document/product/436/12633).

**Notes**

1. If the job to batch restore archived objects includes replicas of objects that have already been restored, the validity period of these objects will be updated when the job starts, ensuring that all object replicas in the job have the same validity period.
2. The batch restore job only initiates requests to restore objects. After all requests are sent, the Batch Operation page will report the job as complete. COS doesn’t notify you when the objects have been restored. However, you can configure event notifications to receive notifications. For more information, see [Event Notifications](https://intl.cloud.tencent.com/document/product/436/12633).
Managing Batch Operation Jobs

This document describes how to manage batch operation jobs in the COS Console.

Filtering Jobs

You can view the list of the batch operation jobs created in the last 90 days using the List Jobs API. The list contains information of each job, such as job ID, description, priority, status, and execution progress. You can filter them by job status in the list to display the ones in the same status or filter them by job description or ID in the console.

Querying Job Status

If you need more information, you can get all the information of a job using the DescribeJob API. This API will return such information as the operation configuration of the specified job, object inventory information, and job report.

Setting Job Priority

You can set priority for your batch operation jobs, and COS will execute them based on priority, i.e., executing jobs of a higher priority first. Priority is expressed as an integer; the greater the value, the higher the priority. You can modify the priority during job execution. If a higher-priority job needs to be added, you can run it first by pausing the lower-priority job.

Although higher-priority jobs generally take precedence over lower-priority ones, priority is not a criterion for sequential execution. If you want to run batch operation jobs sequentially, please monitor the execution status of each job on your own and start them manually.

Job Status
Job status changes as the execution proceeds, as shown below:

The specific meaning of each job status is as follows:

<table>
<thead>
<tr>
<th>Job Status</th>
<th>Description</th>
<th>Next Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>New</td>
<td>When a job is created, its status will be “New”.</td>
<td>It can change to “Preparing”, meaning that COS starts to parse the inventory in the job.</td>
</tr>
<tr>
<td>Preparing</td>
<td>When COS starts to parse the configured inventory in the batch operation job, the job status will change to “Preparing”.</td>
<td>It can change to “Ready” or “Suspended”. The former means that COS has finished parsing the job configuration information and is going to perform the specified operation on the objects in the inventory based on the configuration, while the latter means that COS has finished parsing the job configuration information but is still waiting for your confirmation before it proceeds to perform the specified operation (this status appears if a batch operation job is configured in the console).</td>
</tr>
<tr>
<td>Suspended</td>
<td>A batch operation job pending your confirmation is in the “Suspended” status, which appears if the job is created in the console. After your confirmation, COS will start executing the job, and the status will change accordingly and cannot revert to “Suspended”.</td>
<td>When you confirm to execute the job, the job status will change to “Ready”.</td>
</tr>
<tr>
<td>Ready</td>
<td>When COS has finished parsing the object inventory and job configuration information in your batch operation job and is going to perform the operation, the job status will change to “Ready”.</td>
<td>It can change to “Active”, at which point COS starts executing the job. If a higher-priority job is running, COS will keep the job status as “Ready” until the job status of the higher-priority job changes to “Complete”.</td>
</tr>
<tr>
<td>Job Status</td>
<td>Description</td>
<td>Next Status</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Active</td>
<td>When COS is performing an operation on the objects in the inventory based on the configuration information, the job status will be &quot;Active&quot;. You can query job progress in the console or by calling the DescribeJob API.</td>
<td>It can change to &quot;Complete&quot;, &quot;Failing&quot;, &quot;Pausing&quot;, or &quot;Cancelling&quot; if your job succeeds, fails, or is paused or canceled.</td>
</tr>
<tr>
<td>Pausing</td>
<td>After COS pauses an ongoing job and before the status changes to &quot;Paused&quot;, the job status in between will be &quot;Pausing&quot;.</td>
<td>It can be followed by &quot;Paused&quot; when COS has successfully paused the ongoing job.</td>
</tr>
<tr>
<td>Paused</td>
<td>If a higher-priority job is created, the status of the ongoing job will change to &quot;Paused&quot;.</td>
<td>If the higher-priority job is completed, fails, or is pending conformation, the job in &quot;Paused&quot; status can automatically shift to &quot;Active&quot; status.</td>
</tr>
<tr>
<td>Complete</td>
<td>When the batch operation job has successfully completed the operation on all the objects in the inventory or fails, the job status will change to &quot;Complete&quot;. If job report generation is configured, COS will deliver the report to your specified bucket when the status changes to &quot;Complete&quot;.</td>
<td>&quot;Complete&quot; is a final status, i.e., the job will not change to any other status.</td>
</tr>
<tr>
<td>Cancelling</td>
<td>After COS cancels an ongoing job and before the status changes to &quot;Cancelled&quot;, the job status in between will be &quot;Cancelling&quot;.</td>
<td>It can be followed by &quot;Cancelled&quot; when COS has successfully canceled the ongoing job.</td>
</tr>
<tr>
<td>Cancelled</td>
<td>When a batch operation job is successfully canceled, the job status will change to &quot;Cancelled&quot;. At this point, you cannot make any modification to the job status.</td>
<td>&quot;Cancelled&quot; is a final status, i.e., the job will not change to any other status.</td>
</tr>
<tr>
<td>Failing</td>
<td>The &quot;Failing&quot; status comes before &quot;Failed&quot;.</td>
<td>It can change to the &quot;Failed&quot; status.</td>
</tr>
<tr>
<td>Failed</td>
<td>After a job fails, the job status will change to &quot;Failed&quot;. For more information, please see Tracking a Failing Job.</td>
<td>&quot;Failed&quot; is a final status, i.e., the job will not change to any other status.</td>
</tr>
</tbody>
</table>

Tracking a Failing Job

If a problem arises during job execution, such as trouble in parsing the object inventory, the batch operation job will fail, and COS will return the corresponding error code and cause. You can call the DescribeJob API or view the job report to get the cause of failure and other information.

COS sets an operation failure threshold for each batch operation job to avoid frequent operation failures. If a job involves more than 1,000 operations, COS will monitor the operation failure rate, i.e., ratio of failing operations to all operations executed. If the rate exceeds the threshold of 50% at any moment, COS will terminate the job and return the failure message. You can check the cause why this happens (e.g., the object inventory contains large amounts of information on non-existing objects). Then, you can fix the problem accordingly and create a job again.
COS executes batch operation jobs in an async manner and does not necessarily perform operations in the same order as that of objects in the inventory. Therefore, you cannot determine which object is being operated on according to the order in the object inventory and whether an operation is successful or fails. However, you can get the information on successful or failing operations from the job report.

Job Report

You can configure whether to output a job report when creating a job. If yes, COS will output a job report when the job succeeds, failure, or is canceled. You can view the information on all successful or failing operations in the job report.

A job report contains such information as the configuration parameters and execution status of the specified operation as well as name and version ID of the objects operated on, operation status codes, and error description.
Monitoring and Alarming

Overview

COS statistics such as read and write requests and traffic are collected and displayed by Cloud Monitor. You can view detailed monitoring data in the Cloud Monitor Console.

This document describes how to get statistics in the COS Console. For more information on how to use Cloud Monitor APIs to get more detailed data, see Cloud Monitoring product documentation.

Basic Features

Cloud Monitor provides the following entries for COS to implement monitoring and alarming.

<table>
<thead>
<tr>
<th>Module</th>
<th>Capabilities</th>
<th>Main Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring overview</td>
<td>Displays the current status of the product</td>
<td>Provides general overview, alarm overview, and overall monitoring information</td>
</tr>
<tr>
<td>Alarm management</td>
<td>Supports alarm management and configuration</td>
<td>Supports creating new COS alarming policies, custom messages, and trigger templates</td>
</tr>
<tr>
<td>Monitoring platform</td>
<td>Monitors traffic and displays data of user-defined monitoring metrics</td>
<td>Displays your overall bandwidth information and allows you to customize monitoring metrics and data to be reported</td>
</tr>
<tr>
<td>Cloud product monitoring</td>
<td>Displays the COS bucket monitoring view</td>
<td>Allows you to query the current monitoring views and data such as read and write requests and traffic for each bucket</td>
</tr>
</tbody>
</table>

Use Cases

- **Daily management**: you can log in to the Cloud Monitor Console to view the running status of COS in real time.
- **Troubleshooting**: you will receive alarm notifications when a monitoring metric reaches the alarming threshold, allowing you to quickly get notifications for exceptions, check the causes, and fix them in a timely manner.

Viewing in the Console

You can log in to the Cloud Monitor Console to view the monitoring data for COS.

Calling APIs

You can view the monitoring data for COS by calling the corresponding APIs. Below are the monitoring metrics for COS.

**Storage**

<table>
<thead>
<tr>
<th>Metric Name</th>
<th>Description</th>
<th>Unit</th>
</tr>
</thead>
</table>
### Metric Name | Description | Unit
---|---|---
std_storage | Standard storage - storage capacity | MB
sia_storage | Standard_IA storage - storage capacity | MB
nel_storage | Nearline Storage - storage capacity | MB
arc_storage | Archive storage - storage capacity | MB

### Traffic

<table>
<thead>
<tr>
<th>Metric Name</th>
<th>Description</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>internet_traffic</td>
<td>Public network traffic</td>
<td>B</td>
</tr>
<tr>
<td>internal_traffic</td>
<td>Private network traffic</td>
<td>B</td>
</tr>
<tr>
<td>cdn_origin_traffic</td>
<td>CDN origin-pull traffic</td>
<td>B</td>
</tr>
<tr>
<td>inbound_traffic</td>
<td>Upload traffic</td>
<td>B</td>
</tr>
</tbody>
</table>

### Data Retrieval

<table>
<thead>
<tr>
<th>Metric Name</th>
<th>Description</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>std_retrieval</td>
<td>Standard data retrieval</td>
<td>B</td>
</tr>
<tr>
<td>sia_retrieval</td>
<td>Standard_IA data retrieval</td>
<td>B</td>
</tr>
<tr>
<td>nel_retrieval</td>
<td>Nearline data retrieval</td>
<td>B</td>
</tr>
</tbody>
</table>

### Requests

<table>
<thead>
<tr>
<th>Metric Name</th>
<th>Description</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>std_read_requests</td>
<td>Standard storage read requests</td>
<td>/</td>
</tr>
<tr>
<td>std_write_requests</td>
<td>Standard storage write requests</td>
<td>/</td>
</tr>
<tr>
<td>ia_read_requests</td>
<td>Standard_IA storage read requests</td>
<td>/</td>
</tr>
<tr>
<td>ia_write_requests</td>
<td>Standard_IA storage write requests</td>
<td>/</td>
</tr>
<tr>
<td>nl_read_requests</td>
<td>Nearline storage read requests</td>
<td>/</td>
</tr>
<tr>
<td>nl_write_requests</td>
<td>Nearline storage write requests</td>
<td>/</td>
</tr>
</tbody>
</table>

### Return Codes

<table>
<thead>
<tr>
<th>Metric Name</th>
<th>Description</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>2xx_response</td>
<td>2xx status codes</td>
<td>/</td>
</tr>
<tr>
<td>3xx_response</td>
<td>3xx status codes</td>
<td>/</td>
</tr>
<tr>
<td>4xx_response</td>
<td>4xx status codes</td>
<td>/</td>
</tr>
<tr>
<td>5xx_response</td>
<td>5xx status codes</td>
<td>/</td>
</tr>
</tbody>
</table>
Monitoring Descriptions

- **Monitoring interval**: Cloud Monitor supports multiple types of monitoring intervals, including monitoring data in real time, in the past 24 hours, in the past 7 days, and in user-specified period, with time granularity of 1 minute or 5 minutes.

- **Data retention**: monitoring data with 1-minute, 5-minute, and 1-hour granularity will be retained for 31 days; monitoring data with 1-day granularity will be retained for half a year.

- **Alarm display**: Cloud Monitor integrates the monitoring data of COS and displays the data entries in intuitive graphs, making it easier for you to stay informed of the overall status.

- **Alarm setting**: you can set alarm-triggering thresholds for the metrics. Cloud Monitor will send notifications to specified recipients when the alarm is triggered. For more information, see [https://intl.cloud.tencent.com/doc/product/248/6126).](https://intl.cloud.tencent.com/doc/product/248/6126)
Global Acceleration Overview

Last updated: 2020-03-17 10:56:27

The global acceleration feature of Tencent Cloud COS utilizes a load balancing system based on Tencent's global traffic scheduling to intelligently route and parse user requests and select the optimal network linkage for nearby access. Backed by globally deployed Tencent Cloud data centers, it allows users across the world to quickly access your buckets and improves your business access success rate for a higher business stability and smoother user experience.

At present, the global acceleration function has been launched to support public cloud regions inside and outside China. Financial cloud regions are not supported due to network isolation.

Directions

You can enable global acceleration in the COS Console or through APIs.

Via the COS Console

You can enable global acceleration for your buckets in the COS Console. For more information, please see Enabling Global Acceleration in console documentation.

Via REST API

You can directly use the following APIs to enable global acceleration:

- PUT Bucket Accelerate
- GET Bucket Accelerate

Access domain name

After enabling global acceleration, you can access your COS files through two types of domain names:

- **Default bucket domain name**: The format is `<BucketName-APPID>.cos.<Region>.myqcloud.com`. For more information, please see Regions and Access Domain Names.
- **Global acceleration domain name**: The format is `<BucketName-APPID>.cos.accelerate.myqcloud.com`.

Take the bucket `examplebucket-125000000` in Guangzhou as an example. If you have enabled global acceleration for it, when you need to upload the file `exampleObject.txt` from Beijing to it, you can do so in the following two ways:

- **Use the global acceleration domain name for access**: When uploading the object, you need to set the domain name to `exampleBucket-125000000.cos.accelerate.myqcloud.com`. When you upload the object through this domain name, COS will intelligently parse your request based on your network conditions and implement nearby access. For example, it will forward your request to the Beijing access layer and then transmit it to the Guangzhou storage layer via a private network Direct Connect line to accelerate data transfer.

- **Use the default bucket domain name for access**: When uploading the object, you need to set the domain name to `examplebucket-125000000.cos.ap-guangzhou.myqcloud.com`. When you upload the object through this domain name, your request will be directly forwarded to the Guangzhou access layer and then the Guangzhou storage layer. In this case, long public network linkage may lead to unstable transfer.
If you use global acceleration, fees will be incurred. Therefore, it is recommended that you carefully evaluate this feature based on your actual business needs:

1. If your business has more writes (e.g., PUT Object, POST Object, and Multipart Upload) than reads and uploads data from a remote region to Tencent Cloud data centers, you are recommended to use a global acceleration domain name.
2. If your business has more reads (e.g., GET Object) than writes and mainly involves file download, you are recommended to perform a comprehensive evaluation on the CDN-based access acceleration solution and select the option with the best cost performance.
3. If your business mainly involves configuration operations or file extraction, you are recommended to use a default bucket domain name.
4. If your business needs to access buckets over the private network or a direct connect line in the same region, you are recommended to use a default bucket domain name.

Notes

Below are precautions for using a global acceleration domain name:

- The global acceleration domain name will take effect in 15 minutes after being enabled. Please wait patiently.
- After the global acceleration domain name is enabled, the maximum bandwidth for a bucket through it will be allocated based on the business volume on the entire network.
- After the global acceleration domain name is enabled, only requests with it will be accelerated. However, the default bucket domain name can still be used.
- When the global acceleration domain name is used, fees will be incurred only for the accelerated linkage of requests. For example, if you use the global acceleration domain name to upload data from Beijing to a bucket in Beijing, as the linkage is not accelerated, the request will not incur acceleration fees.
- When using the global acceleration domain name, you can specify the HTTP or HTTPS transfer protocol. If the request is transmitted via a private network Direct Connect line, COS will choose to use the HTTPS protocol to guarantee data transfer security as appropriate.
Data Processing

Overview of picture processing

Last updated: 2020-04-27 22:36:38

Overview

Tencent Cloud COS has integrated the professional All-in-One media solutions of Cloud Infinite, covering image processing, moderation, recognition, and many more. See the table below for details. You can process your media data with the Upload and Process APIs in COS.

<table>
<thead>
<tr>
<th>Service</th>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic image processing service</td>
<td>Scaling</td>
<td>Proportional scaling, scale an image based on the target width and height, and more</td>
</tr>
<tr>
<td></td>
<td>Cropping</td>
<td>Cut (regular cropping), crop (scaling and cropping), iradius (inscribed circle cropping), and scrop (smart cropping)</td>
</tr>
<tr>
<td></td>
<td>Rotation</td>
<td>Adaptive rotation and common rotation</td>
</tr>
<tr>
<td></td>
<td>Format conversion</td>
<td>Format conversion, GIF format optimization, and progressive display</td>
</tr>
<tr>
<td></td>
<td>Quality conversion</td>
<td>Perform quality conversion on images in JPG and WEBP formats</td>
</tr>
<tr>
<td></td>
<td>Gaussian blurring</td>
<td>Blur images</td>
</tr>
<tr>
<td></td>
<td>Sharpening</td>
<td>Sharpen images</td>
</tr>
<tr>
<td></td>
<td>Watermarking</td>
<td>image watermarks, text watermarks</td>
</tr>
<tr>
<td></td>
<td>Obtaining image information</td>
<td>Basic information, EXIF information, average hue</td>
</tr>
<tr>
<td></td>
<td>Removing meta information</td>
<td>Including EXIF information</td>
</tr>
<tr>
<td></td>
<td>Quick thumbnail template</td>
<td>Perform quick format conversion, scaling, and cropping to generate thumbnails</td>
</tr>
<tr>
<td></td>
<td>Setting styles</td>
<td>Set image styles to manage images for different purposes</td>
</tr>
<tr>
<td>Value-added image services</td>
<td>Blind watermarking</td>
<td>Add an invisible watermark to the original image without affecting its quality</td>
</tr>
<tr>
<td></td>
<td>Content detection</td>
<td>Detect obscene, political, violence, and advertising content</td>
</tr>
<tr>
<td></td>
<td>QR code recognition</td>
<td>Recognize and override (if necessary) the position and content of a valid QR code in an image</td>
</tr>
<tr>
<td></td>
<td>Image tag</td>
<td>Identify the main content in an image and return its tag information</td>
</tr>
</tbody>
</table>

Directions

Using the COS Console

You can set image processing in the COS Console. For more information, please see Enabling Image Processing.
Using REST API

You can configure image processing with the Cloud Infinite API. For more information, see the Cloud Infinite API documentation Basic Image Processing.