

Cloud Object Storage

User Tools

Product Documentation



Copyright Notice

©2013-2025 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 the Tencent corporate group, including its parent, subsidiaries and affiliated companies, as the case may be. 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

User Tools

- Tool Overview

- Installation and Configuration of Environment

 - Java

 - Python

 - Hadoop

COSBrowser

- COSBrowser Overview

- User Guide for Desktop Version

- User Guide for Mobile Version

 - Installation and Login

 - Mobile Version Features

 - Bucket Management and Operations

 - File Management and Operations

 - Data Monitoring

COSCLI (Beta)

- COSCLI Overview

- Download and Installation Configuration

- Common Options

- Common Commands

 - Generating and Modifying Configuration Files - config

 - Creating Buckets - mb

 - Deleting Buckets - rb

 - Tagging Bucket - bucket-tagging

 - Querying Bucket/Object List - ls

 - Obtaining Statistics on Different Types of Objects - du

 - Uploading/Downloading/Copying Objects - cp

 - Syncing Upload/Download/Copy - sync

 - Deleting Objects - rm

 - Getting File Hash Value - hash

 - Listing Incomplete Multipart Uploads - lsparts

 - Clearing Incomplete Multipart Uploads - abort

 - Retrieving Archived Files - restore

 - Creating/Obtaining a Symbolic Link - symlink

 - Viewing Contents of an Object - cat

[Getting Pre-signed URL - signurl](#)

[Listing Contents and Statistics Under a Directory - lsdu](#)

[Bucket Versioning - bucket-versioning](#)

[FAQs](#)

[COSCMD](#)

[COS Migration](#)

[FTP Server](#)

[Hadoop](#)

[COSDistCp](#)

[HDFS TO COS](#)

[GooseFS-Lite](#)

[Online Auxiliary Tools](#)

[COS Request Tool](#)

[Diagnostic Tool](#)

User Tools

Tool Overview

Last updated : 2024-01-06 16:15:35

Tool	Features
COS Browser	This tool makes it easy for users to perform data upload/download, access link generation, and other operations in a visualized manner.
COSCLI	COS provides the command-line client COSCLI to allow you to upload, download, delete, and perform other operations on COS objects by using simple commands.
COSCMD	Enables users to perform operations (such as upload, download, and delete) in batches with simple commands.
COS Migration	This tool is used to migrate data from multiple data sources (such as on-premises server, and other cloud storage services) to COS.
FTP Server	Enables users to upload/download files to/from COS by using an FTP client.
COSFS	In Linux, this tool is used to mount buckets to a local file system and to perform operations on objects in COS via the local file system.
Hadoop Tool	Helps integrate COS with big data computing frameworks such as Hadoop, Spark, and Tez, so that they can read and write COS data.
COSDistCp	COSDistCp is a MapReduce-based distributed file copy tool mainly used for data copy between HDFS and COS.
Hadoop-cos-DistChecker	Verifies the directory integrity after you use the <code>hadoop distcp</code> command to migrate data from HDFS to COS.
HDFS TO COS	Copies data from HDFS to COS.
Diagnostic Tool	COS's web-based diagnostic tool that allows you to troubleshoot error requests.

Upload Capabilities

The upload capabilities of different tools are detailed as below:

Tool	Simple Upload	Multipart Upload	Checkpoint Restart	Advanced Upload	Consistency Check	Pre-Signed URL Generation

COSBrowser	Supported	Supported	Supported	Supported. By default, multipart upload will be triggered for files of 8 MB or above.	Supported for MD5 check	Supported for file download
COSCLI	Supported	Supported	Supported	Supported. The trigger threshold for multipart upload can be customized.	Supported for CRC64 check	Supported
COSCMD	Supported	Supported	Supported	Supported. By default, multipart upload will be triggered for files of 10 MB or above.	Supported for MD5 check	Supported
COS Migration	Supported	Supported	Supported	Supported. The trigger threshold for multipart upload can be customized.	Supported for MD5 check	N/A
FTP Server	Supported	Supported	Supported	Supported. The trigger threshold for multipart upload can be customized.	Not supported	N/A
COSFS	Supported	Supported	Supported	Supported. The trigger threshold for multipart	Supported for MD5 check	N/A

				upload can be customized.		
Hadoop	Supported	Supported	Not supported. There is an HDFS protocol conflict.	Supported. The trigger threshold for multipart upload can be customized.	Supported for CRC64 or CRC32 check	N/A
COSDistCp	Supported	Supported	Not supported. There is an HDFS protocol conflict.	Supported. The trigger threshold for multipart upload can be customized.	Supported for file size, CRC64, or CRC32 check	N/A
HDFS TO COS	Supported	Supported	Not supported. There is an HDFS protocol conflict.	Supported. The trigger threshold for multipart upload can be customized.	Supported for file name or size check	N/A

Note:

Advanced upload encapsulates simple upload and multipart upload and can intelligently select the upload method based on file size.

Installation and Configuration of Environment Java

Last updated : 2024-01-06 16:15:34

Java Development Kit (JDK) is the SDK for Java. This document takes JDK 1.7 and 1.8 as examples to describe how to install and configure JDK under Windows and Linux systems.

Windows

1. Downloading a JDK

Go to the [Oracle website](#) to download the desired JDK version.

2. Installation

Install the JDK as instructed. You can specify the installation paths (C drive by default), for example, as

`D:\Program Files\Java\jdk1.8.0_31` and `D:\Program Files\Java\jre1.8.0_31` .

3. Configuration

After the installation is completed, right-click **Computer**, and then click **Properties > Advanced system settings > Environment Variables > System variables > New** to configure the software.

Variable name (N): **JAVA_HOME** Variable value (V): `D:\Program Files\Java\jdk1.8.0_31` (Configure according to your actual installation path).

Variable name (N): **CLASSPATH** Variable value (V):

`.;%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar;` (Note that the variable value begins with `.`).

Variable name (N): **Path**

Variable value (V): `%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin;`

4. Testing

Test whether the configuration is successful: click **Start** (or shortcut: Win+R) > **Run** (enter `cmd`) > **OK** (or press Enter), then enter the command `javac` and press Enter. If messages such as command parameters and syntax are displayed, the environment variables are configured successfully.

Linux

If openjdk is installed by using yum or apt-get command, the class library may be incomplete, thus leading to errors when you run relevant tools after the installation. Therefore, we recommend that you manually decompress and install JDK. Specific steps are as follows:

1. Download a JDK

Go to the [Oracle website](#) to download the desired JDK version to install.

Note:

The following uses `jdk-8u151-linux-x64.tar.gz` as an example. If you are using other versions, ensure that the extension is `.tar.gz`.

2. Create a directory

Run the following command to create the `java` directory in `/usr/`:

```
mkdir /usr/java
cd /usr/java
```

Copy the downloaded `jdk-8u151-linux-x64.tar.gz` to the `/usr/java/` directory.

3. Decompress the JDK

Run the following command to decompress the JDK:

```
tar -zxvf jdk-8u151-linux-x64.tar.gz
```

4. Set environment variables

Edit the `/etc/profile` file. Add the following content to the profile file and save it:

```
# set java environment
JAVA_HOME=/usr/java/jdk1.8.0_151
JRE_HOME=/usr/java/jdk1.8.0_151/jre
CLASS_PATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar:$JRE_HOME/lib
PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin
export JAVA_HOME JRE_HOME CLASS_PATH PATH
```

Note:

`JAVA_HOME` and `JRE_HOME` should be configured according to the actual installation paths and JDK version. Run the following command for the modifications to take effect:

```
source /etc/profile
```

5. Test

Run the following command to test the JDK installation:

```
java -version
```

If information about the Java version is displayed, the JDK is installed successfully.

```
java version "1.8.0_151"  
Java(TM) SE Runtime Environment (build 1.8.0_151-b12)  
Java HotSpot(TM) 64-Bit Server VM (build 25.151-b12, mixed mode)
```

Python

Last updated : 2024-01-06 16:15:34

This document describes how to install Python for different operating systems.

Using an Installation Package

1. Download a package

Go to the [Python website](#) to download an installation package according to your OS.

Note:

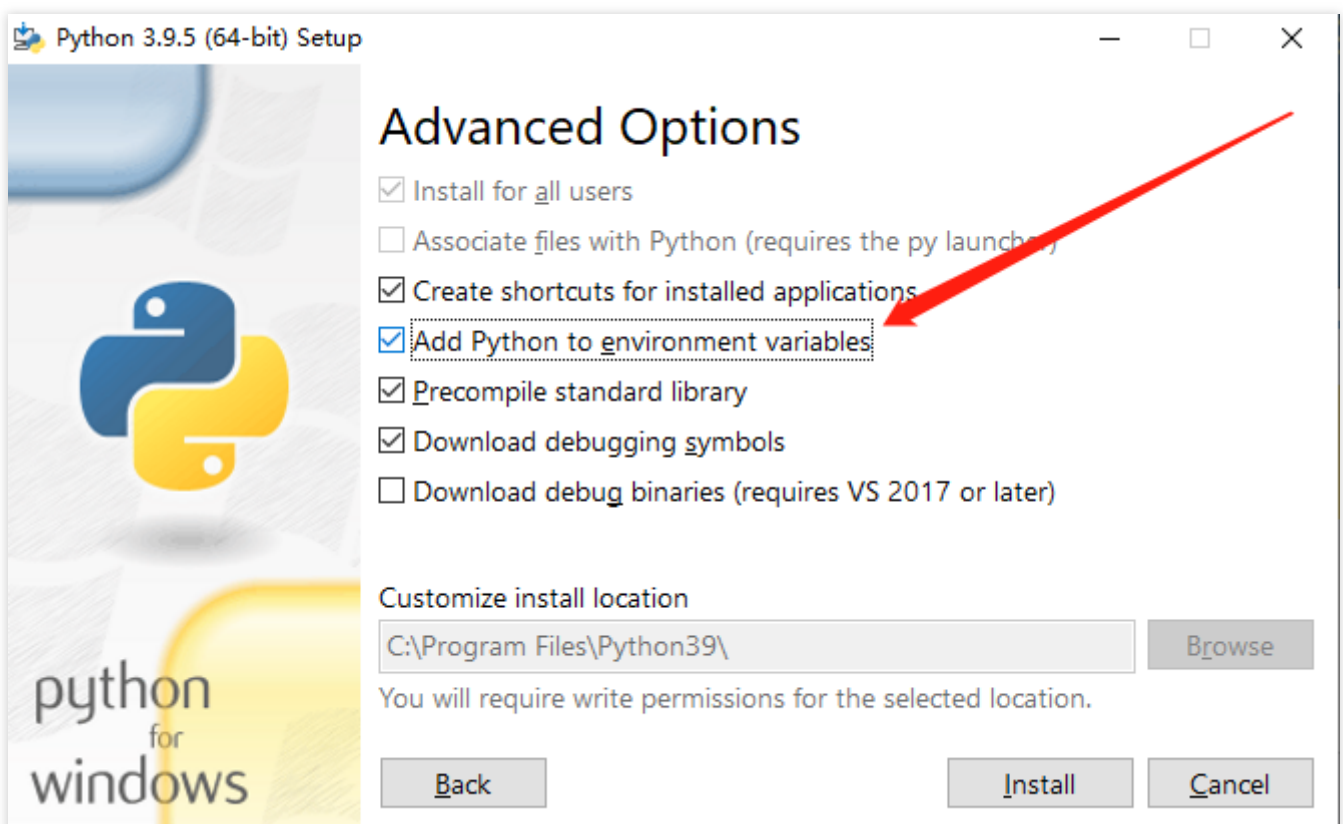
Python has dropped support for Python 2 since January 1, 2020. Therefore, you are advised to install Python 3.

2. Install the package

Install the downloaded package as instructed.

Note:

If you use Windows, check **Add Python to environment variables**.



3. Verify the installation

Run the following command in Terminal to view the Python version:

```
python -V
```

If the Python version is displayed, Python has been installed successfully.

Note:

If you use Windows, you may need to restart your computer after the installation.

4. Configure environment variables

In Windows, if “not recognized as an internal or external command” is reported in Terminal after the command above is run, right-click the **Computer** icon, click **Properties > Advanced system settings > Environment Variables**, and in the **System variables** area, click **New** to add the Python installation path:

Using a Package Manager

macOS

Install [HomeBrew](#) and use it to install Python:

```
brew install python
```

Ubuntu

Use the built-in Advanced Packaging Tool (APT) to install Python:

```
sudo apt-get install python
```

CentOS

Use the built-in Yellowdog Updater, Modified (YUM) to install Python:

```
sudo yum install -y python
```

Hadoop

Last updated : 2025-01-24 13:06:55

Hadoop (2.7.2 or above) tool provides the capability to run computing tasks using Tencent Cloud COS as the underlying file storage system. The Hadoop cluster can be launched in three modes: stand-alone, pseudo-distributed, and fully-distributed. This document uses Hadoop-2.7.4 as an example to describe how to build a fully-distributed Hadoop environment and how to use wordcount to execute a simple test.

Preparation

Prepare several servers.

Install and configure the system by downloading from the [CentOS Official Website](#). This document uses CentOS 7.3.1611.

Install the Java environment. For more information, see [Installing and Configuring Java](#).

Install the available Hadoop package: [Apache Hadoop Releases Download](#).

Network Configuration

Use `ifconfig -a` to check the IP of each server, then use the ping command to check if they can ping each other, and record the IP of each server.

Configuring CentOS

Configure a Hostname

Set the corresponding hostname for each machine, such as "master", "slave*", etc.

```
hostnamectl set-hostname master
```

Configure hosts

```
vi /etc/hosts
```

Edit the content:

```
202.xxx.xxx.xxx master
202.xxx.xxx.xxx slave1
202.xxx.xxx.xxx slave2
202.xxx.xxx.xxx slave3
# Replace IPs with the real ones
```

Turn off firewall

```
systemctl status firewalld.service # Check firewall status
systemctl stop firewalld.service # Turn off firewall
systemctl disable firewalld.service # Disable firewall to start on boot
```

Time synchronization

```
yum install -y ntp # Install ntp service
ntpdate cn.pool.ntp.org # Sync network time
```

Install and configure JDK

Upload JDK installer package (such as `jdk-8u144-linux-x64.tar.gz`) to the `root` directory.

```
mkdir /usr/java
tar -zxvf jdk-8u144-linux-x64.tar.gz -C /usr/java/
rm -rf jdk-8u144-linux-x64.tar.gz
```

Copy JDKs among hosts

```
scp -r /usr/java slave1:/usr
scp -r /usr/java slave2:/usr
scp -r /usr/java slave3:/usr
.....
```

Configure environment variables for JDK of each host

```
vi /etc/profile
```

Edit the content:

```
export JAVA_HOME=/usr/java/jdk1.8.0_144
export PATH=$JAVA_HOME/bin:$PATH
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
```

After saving the file, make `/etc/profile` take effect by executing the following command:

```
source /etc/profile # Make the configuration file effective
java -version # View java version
```

Configuring Keyless Access via SSH

Check the SSH service status on each host:

```
systemctl status sshd.service # Check the SSH service status.
yum install openssh-server openssh-clients # Install the SSH service. Ignore
this step if it is already installed.
systemctl start sshd.service # Enable the SSH service. Ignore this step if it
is already enabled.
```

Generate a key on each host:

```
ssh-keygen -t rsa # Generate Keys
```

On slave1:

```
cp ~/.ssh/id_rsa.pub ~/.ssh/slave1.id_rsa.pub
scp ~/.ssh/slave1.id_rsa.pub master:~/.ssh
```

On slave2:

```
cp ~/.ssh/id_rsa.pub ~/.ssh/slave2.id_rsa.pub
scp ~/.ssh/slave2.id_rsa.pub master:~/.ssh
```

And so on...

On master:

```
cd ~/.ssh
cat id_rsa.pub >> authorized_keys
cat slave1.id_rsa.pub >>authorized_keys
cat slave2.id_rsa.pub >>authorized_keys
scp authorized_keys slave1:~/.ssh
scp authorized_keys slave2:~/.ssh
scp authorized_keys slave3:~/.ssh
```

Installing and Configuring Hadoop

Installing Hadoop

Upload the Hadoop installer package (such as `hadoop-2.7.4.tar.gz`) to the `root` directory.

```
tar -zxvf hadoop-2.7.4.tar.gz -C /usr
rm -rf hadoop-2.7.4.tar.gz
mkdir /usr/hadoop-2.7.4/tmp
mkdir /usr/hadoop-2.7.4/logs
mkdir /usr/hadoop-2.7.4/hdf
mkdir /usr/hadoop-2.7.4/hdf/data
```

```
mkdir /usr/hadoop-2.7.4/hdf/name
```

Go to the `hadoop-2.7.4/etc/hadoop` directory and proceed to the next step.

Configure Hadoop

1. Add the following to the `hadoop-env.sh` file.

```
export JAVA_HOME=/usr/java/jdk1.8.0_144
```

If the SSH port is not 22 (default value), modify it in the `hadoop-env.sh` file:

```
export HADOOP_SSH_OPTS="-p 1234"
```

2. Modify `yarn-env.sh`

```
export JAVA_HOME=/usr/java/jdk1.8.0_144
```

3. Modify `slaves`

Configure the content:

```
Delete:
localhost
Add:
slave1
slave2
slave3
```

4. Modify `core-site.xml`

```
<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://master:9000</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>file:/usr/hadoop-2.7.4/tmp</value>
  </property>
</configuration>
```

5. Modify `hdfs-site.xml`

```
<configuration>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>/usr/hadoop-2.7.4/hdf/data</value>
    <final>true</final>
  </property>
</configuration>
```



```
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>/usr/hadoop-2.7.4/hdf/name</value>
  <final>true</final>
</property>
</configuration>
```

6. Copy `mapred-site.xml.template` and name it `mapred-site.xml`

```
cp mapred-site.xml.template mapred-site.xml
```

7. Modify `mapred-site.xml`

```
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
  <property>
    <name>mapreduce.jobhistory.address</name>
    <value>master:10020</value>
  </property>
  <property>
    <name>mapreduce.jobhistory.webapp.address</name>
    <value>master:19888</value>
  </property>
</configuration>
```

8. Modify `yarn-site.xml`

```
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
    <value>org.apache.mapred.ShuffleHandler</value>
  </property>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.resourcemanager.address</name>
    <value>master:8032</value>
  </property>
  <property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>master:8030</value>
  </property>
</configuration>
```

```
</property>
<property>
  <name>yarn.resourcemanager.resource-tracker.address</name>
  <value>master:8031</value>
</property>
<property>
  <name>yarn.resourcemanager.admin.address</name>
  <value>master:8033</value>
</property>
<property>
  <name>yarn.resourcemanager.webapp.address</name>
  <value>master:8088</value>
</property>
</configuration>
```

9. Copy Hadoop among hosts

```
scp -r /usr/ hadoop-2.7.4 slave1:/usr
scp -r /usr/ hadoop-2.7.4 slave2:/usr
scp -r /usr/ hadoop-2.7.4 slave3:/usr
```

10. Configure environment variables for Hadoop of each host

Open the configuration file:

```
vi /etc/profile
```

Edit the content:

```
export HADOOP_HOME=/usr/hadoop-2.7.4
export PATH=$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$PATH
export HADOOP_LOG_DIR=/usr/hadoop-2.7.4/logs
export YARN_LOG_DIR=$HADOOP_LOG_DIR
```

Implement the configuration file:

```
source /etc/profile
```

Start Hadoop

1. Format namenode

```
cd /usr/hadoop-2.7.4/sbin
hdfs namenode -format
```

2. Start

```
cd /usr/hadoop-2.7.4/sbin
```

```
start-all.sh
```

3. Check processes

If processes on master contain ResourceManager, SecondaryNameNode and NameNode, Hadoop starts successfully. For example:

```
2212 ResourceManager
2484 Jps
1917 NameNode
2078 SecondaryNameNode
```

If processes on each slave contain DataNode and NodeManager, Hadoop starts successfully. For example:

```
17153 DataNode
17334 Jps
17241 NodeManager
```

Running wordcount

The wordcount built in Hadoop can be called directly. After Hadoop starts, use the following command to work with files in HDFS:

```
hadoop fs -mkdir input
hadoop fs -put input.txt /input
hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.4.jar wordcount
/input /output/
```

```
root@VM-96-24-centos /usr/hadoop-2.7.4# hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.4.jar wordcount /input /output/
17/07/18 23:04:51 INFO client.RMProxy: Connecting to ResourceManager at master/10.104.96.24:8032
17/07/18 23:04:53 INFO input.FileInputFormat: Total input paths to process : 1
17/07/18 23:04:53 INFO mapreduce.JobSubmitter: number of splits:1
17/07/18 23:04:54 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1500344813707_0002
17/07/18 23:04:54 INFO impl.YarnClientImpl: Submitted application application_1500344813707_0002
17/07/18 23:04:54 INFO mapreduce.Job: The url to track the job: http://master:8088/proxy/application_1500344813707_0002/
17/07/18 23:04:54 INFO mapreduce.Job: Running job: job_1500344813707_0002
17/07/18 23:05:01 INFO mapreduce.Job: Job job_1500344813707_0002 running in uber mode : false
17/07/18 23:05:01 INFO mapreduce.Job: map 0% reduce 0%
17/07/18 23:05:05 INFO mapreduce.Job: map 100% reduce 0%
17/07/18 23:05:11 INFO mapreduce.Job: map 100% reduce 100%
17/07/18 23:05:12 INFO mapreduce.Job: Job job_1500344813707_0002 completed successfully
17/07/18 23:05:12 INFO mapreduce.Job: Counters: 49
```

The above result shows that Hadoop is installed successfully.

View output directory

```
hadoop fs -ls /output
```

View output result

```
hadoop fs -cat /output/part-r-00000
```

```
[root@VM_96_24_centos /usr/hadoop-2.7.4]# hadoop fs -cat /output/part-r-00000
a          5
dasdada    1
ds         2
qwe        1
ret        1
s          1
v          2
vdfd       1
wqere      1
```

Note:

For more information on how to run Hadoop in stand-alone and pseudo-distributed modes, see [Get Started with Hadoop](#) on the official website.

COSBrowser

COSBrowser Overview

Last updated : 2024-04-17 15:03:28

COSBrowser is a visualization interface tool provided by Tencent Cloud COS. You can use it to view, transfer, and manage COS resources easily. Currently, it is available for desktop and mobile clients as described below. If you need to migrate or batch upload data, you can use [Migration Service Platform \(MSP\)](#).

[User Guide for Desktop Version](#)

[User Guide for Mobile Version](#)

Download Address

COSBrowser	OS	System Requirements	Download Address
Desktop	Windows	Windows 7 32/64-bit or later, Windows Server 2008 R2 64-bit or later	Windows
	macOS	macOS 10.13 or later	macOS Intel Chip Edition macOS Apple Chip Edition
	Linux	Includes a GUI that supports the ApplImage format Note: To launch a client that runs CentOS, you need to run <code>./cosbrowser.ApplImage --no-sandbox</code> in the terminal.	Linux
Mobile Version	Android	Android 4.4 or later	Android
	iOS	iOS 11 or above	iOS
Web	Web	Browsers such as Chrome, Firefox, Safari, and Internet Explorer 10+	Web

COSBrowser Desktop Version

COSBrowser Desktop Version focuses on resource management and uploading and downloading data in batches.

Note:

COSBrowser Desktop Version uses the system-configured proxy to connect to the internet. Make sure that your proxy is set up properly, or you can disable the proxy configuration if it fails to connect to the internet.

For queries on Windows, go to **Internet Options**.

For queries on macOS, go to **Network Preferences**.

For queries on Linux, go to **System Settings > Network > Network Proxy**.

COSBrowser Desktop Version has the following features:

Feature	Description
Creating/Deleting a bucket	Creates or deletes a bucket.
Viewing bucket details	Views the basic information of your bucket.
Viewing statistics	Views the current storage capacity and number of objects in your bucket.
Permission management	Modifies the permissions on your buckets and objects.
Setting versioning	Enables/Suspends bucket versioning.
Adding an access path	Adds an access path.
Uploading files/folders	Uploads files/folders to a bucket separately, in batches, or incrementally. Note: 1. Up to 100,000 files can be batch uploaded in one request. 2. Checkpoint restart is not supported. 3. To migrate or batch upload data, use Migration Service Platform (MSP) .
Downloading a file/folder	Downloads files/folders to the local file system separately, in batches, or incrementally. Note: 1. Up to 100,000 files can be batch uploaded in one request. 2. Checkpoint restart is not supported.
Deleting a file/folder	Deletes files/folders from a bucket separately or in batches.
Synchronizing files	Synchronizes local files to your bucket in real time.
Copying and pasting files	Copies files/folders separately or in batches from one directory to another.
Renaming files	Renames files in a bucket.
Creating a folder	Creates a folder in a bucket.

Viewing file details	Views the basic information of the files in a bucket.
Generating a file link	Generates a file access link with a certain validity period by requesting a temporary signature.
Sharing a file/folder	Shares a file/folder and sets a validity period for the sharing.
Exporting file URLs	Exports file URLs in batches.
Previewing a file	Previews media files (images, video, and audio) in your bucket.
Searching a file	Searches files in a bucket through prefix search.
Searching buckets	Searches existing buckets.
Viewing file versions/incomplete multipart uploads	Views multiple versions of a file in a versioning-enabled bucket. Views the incomplete multipart uploads in your bucket.
Comparing files	Compares files in a local folder to those in a bucket.
Transcoding a video	Transcodes videos with the media processing feature enabled in a bucket.
Generating an authorization code	Generates an authorization code for logging in to the COSBrowser client.
Processing an image	Scales, crops, or rotates an image, or adds text or image watermarks, and generates a URL of the output image.
Setting up a proxy	Sets up a proxy to access COS.
Setting the number of concurrent uploads/downloads	Sets the number of concurrent transfers for file upload or download.
Setting the number of parts to upload/download	Sets the number of parts for multipart upload or download.
Setting the number of retries upon upload/download failure	Sets the number of retries upon upload or download failure.
Limiting single-thread upload/download speed	Limits the upload and download speeds for a single thread.
Setting upload check	Double-checks files uploaded to a bucket.
Viewing a local log	Saves the record of operations on COSBrowser in the form of a local log.

COSBrowser Mobile Version

The COSBrowser mobile version is mainly used to monitor COS resources (such as storage usage and traffic) at any time you want. For more supported features, see basic features in [User Guide for Mobile Version](#).

Changelog

Desktop Version changelog: [changelog](#).

Mobile Version changelog: [changelog_mobile](#).

Feedback and Suggestions

If you have any questions or suggestions during your use of COSBrowser, please feel free to give us your feedback:

Feedback on Desktop Version: [issues](#).

Feedback on Mobile Version: [issues_mobile](#).

User Guide for Desktop Version

Last updated : 2024-04-17 14:40:00

Download and Installation

Downloads

OS	OS Requirement	Download Link
Windows	Windows 7 32/64-bit or above, Windows Server 2008 R2 64-bit or above	Windows
macOS	macOS 10.13 or above	macOS Intel Chip Edition macOS Apple Chip Edition
Linux	Includes GUI and supports the AppImage format	Linux
Web	Browsers such as Chrome, Firefox, Safari, and Internet Explorer 10+	Web

Installations

You can install the tool by using the installer or decompressing the installation package for your platform. You can also directly use it in browsers without installation.

Note:

To launch the client running CentOS, you need to run `./cosbrowser.AppImage --no-sandbox` in the terminal.

Login

You can log in to the COSBrowser desktop version using a permanent key, Tencent Cloud account, or a shared link. You can log in to an account on multiple devices at the same time.

Login with a permanent key

You can log in using your Tencent Cloud API key (`SecretID` and `SecretKey`), which can be created and obtained at [Manage API Key](#) in the CAM console. After you logged in successfully, the key will be saved to **Historical Sessions**. The login page and configuration items are as follows:

Bucket/Access Path: If the root account allows you to access only a specific bucket or directory with the currently used key, then this configuration item is required. After setting it, you can quickly enter the corresponding file path. The

path format is `Bucket` or `Bucket/Object-prefix` ; for example, if you are allowed to access only the `doc` folder in the `examplebucket-1250000000` bucket with the current key, enter `examplebucket-1250000000/doc` .

Description: description of the permanent key entered, such as the operator and the usage. The description can be used to distinguish different `SecretID` when you manage the historical sessions on the historical key page.


Remember Session :

If this box is not checked, the Tencent Cloud API key entered will be cleared when you log out (if the key has been saved to the historical sessions, it will be removed).

If this box is checked, the Tencent Cloud API key entered will be remembered and can be managed in the historical sessions.

Note:

You cannot log in to COSBrowser with a project key.

The image shows the COSBrowser Desktop Version interface. On the left, there is a large blue graphic with the COSBrowser logo at the top. Below the logo, there is a stylized illustration of a laptop displaying a web interface, with a large blue arrow pointing downwards from the laptop towards a white document icon at the bottom. The right side of the interface is a white panel titled 'Key Login'. It contains several input fields: 'SecretID' with the value 'AKID7wXsRnJIAI8G7hVzMSHsDH4v...', 'SecretKey' with a masked value '*****', 'Buckets/Access Path' with the value 'examplebucket-1250000000/test/', 'Region' with a dropdown menu showing 'ap-chengdu', and 'Remark' with the text 'Not required, Add remark'. Below these fields is a blue 'Login' button. At the bottom of the panel, there are three links: 'Get SecretKey', 'History key', and 'Local Logs'.

Key Login [Advanced Setting >](#)

SecretID

AKID7wXsRnJIAI8G7hVzMSHsDH4v...

SecretKey

Buckets/Access Path ⓘ

examplebucket-1250000000/test/

Region

ap-chengdu ▼

Remark

Not required, Add remark

Login

[Get SecretKey](#) [History key](#) [Local Logs](#)

Login with a Tencent Cloud account

Click **Login with Tencent Cloud Account** and use your Tencent Cloud account to log in to COSBrowser Desktop Version in the pop-up window. You can log in to your Tencent Cloud account via email or sub-user. For detailed directions, see [About Account](#).

Login with a shared link

You can log in to COSBrowser Desktop Version temporarily through the **shared link** and **password** forwarded or shared by another user. For more information, see [Sharing file/folder](#).

Basic Features

Note:

COSBrowser for Windows v2.11.1 is used as an example here. For other versions, see [Changelog](#).

1. Creating/Deleting bucket

Feature	Description	Directions
Creating bucket	You can create a bucket directly via the client	<ol style="list-style-type: none">1. In the bucket list, click Add Bucket in the upper-left corner.2. Enter the bucket name correctly, and select your region and access permissions. You can also set bucket tags as needed.3. Click OK.
Deleting bucket	Before you delete a bucket, ensure that all data in the bucket has been cleared	<ol style="list-style-type: none">1. In the bucket list, click Delete on the right of the selected bucket.2. Click OK.

2. Viewing bucket details

You can view bucket details by clicking **Details** on the right of the bucket list. Details include bucket name, region, access permissions, MAZ configuration, versioning status, global acceleration status, and bucket domain name.

Note:

The MAZ configuration currently is only supported in certain regions, such as Beijing, Guangzhou, Shanghai, and Singapore. For more information, see [MAZ Feature Overview](#).

3. Viewing statistics

You can click ... > **Statistics** to view the bucket statistics, including the storage usage and the number of objects.

4. Managing permissions

You can use COSBrowser to manage permissions for buckets and objects.

Bucket permissions: Click **Permission Management** on the right of the bucket list.

Object permissions: Click **Permission Management** on the right of the object.

Note:

For more information about COS permissions, please see [ACL Overview](#).

5. Setting versioning

You can use COSBrowser to enable/disable versioning for a bucket.

Click **Details** in the **Operation** column on the right of the bucket list. In the bucket details pop-up window, click the versioning icon.

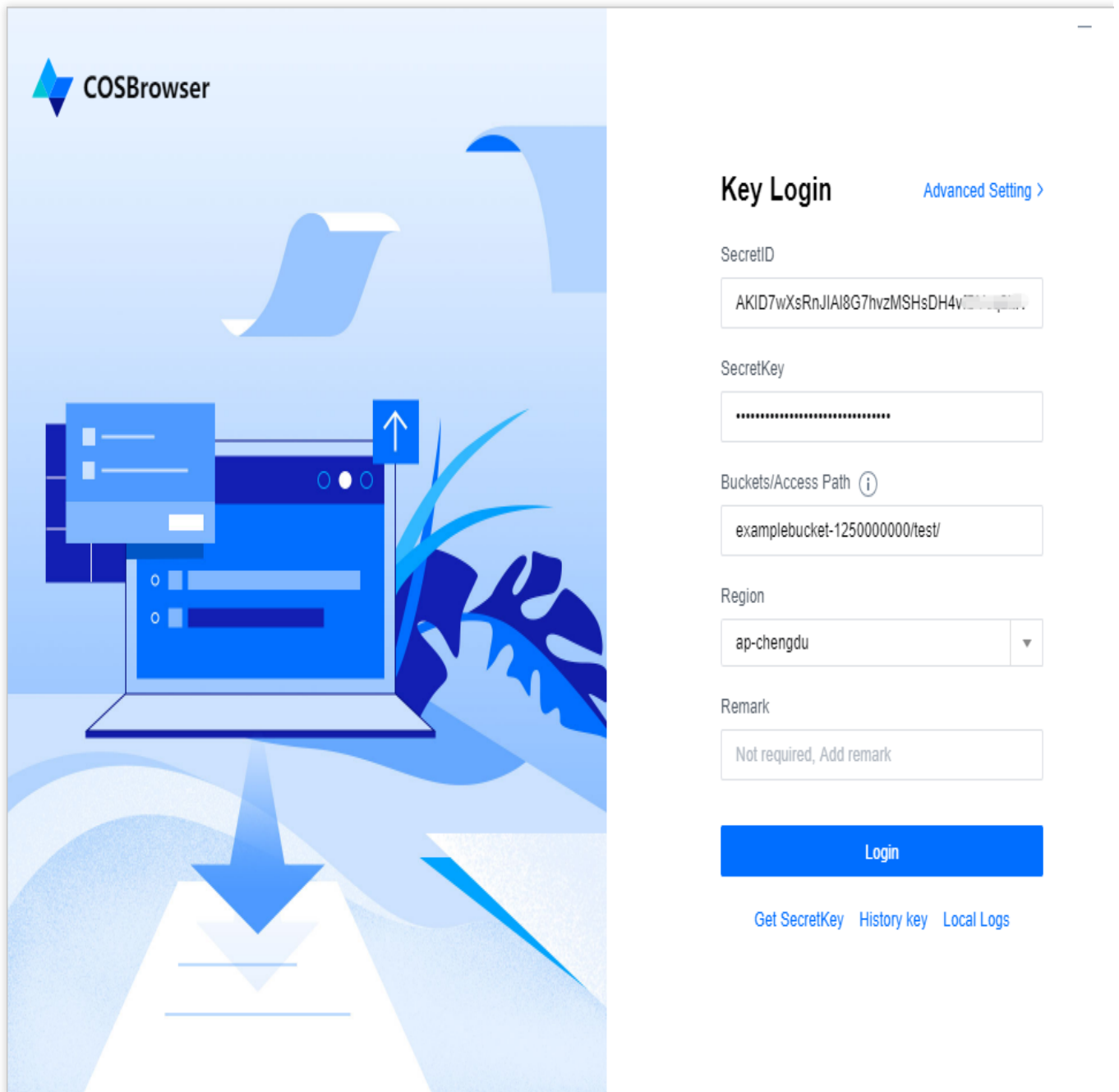
Note:

For more information about versioning, see [Versioning Overview](#).

6. Adding an access path

If you log in with a sub-account that does not have permission to access the bucket list, you can initiate an access via **Add Access Path** in the following two ways:

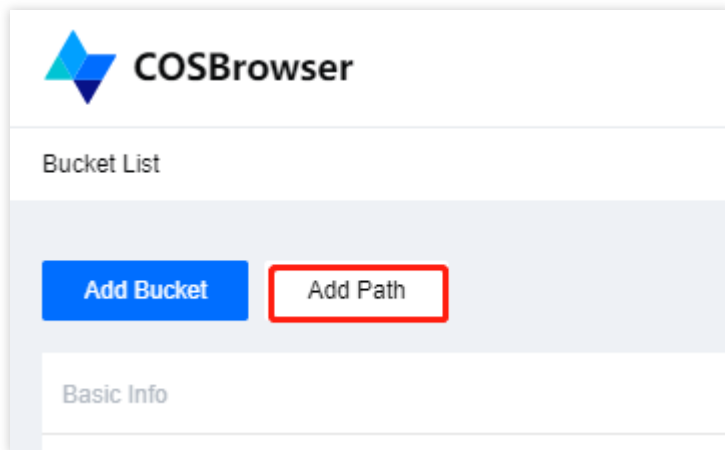
(1) Add an access path directly on the login page and select the corresponding bucket region. Once you log in, you can manage your resources.



The image shows the COSBrowser Key Login interface. On the left is a large blue illustration featuring a laptop with a web browser interface, a large blue arrow pointing downwards, and stylized clouds and plants. The COSBrowser logo is in the top left of this illustration. On the right is the login form with the following fields and options:

- Key Login** (with a link to [Advanced Setting >](#))
- SecretID**: A text box containing the value `AKID7wXsRnJIAI8G7hVzMSHsDH4v`.
- SecretKey**: A text box filled with dots, indicating a masked password.
- Buckets/Access Path** (with an information icon *i*): A text box containing the value `examplebucket-1250000000/test/`.
- Region**: A dropdown menu with the selected value `ap-chengdu`.
- Remark**: A text box containing the value `Not required, Add remark`.
- Login**: A large blue button.
- Below the Login button are three links: [Get SecretKey](#), [History key](#), and [Local Logs](#).

(2) Log in with your sub-account, click **Add Path** in the upper-left corner of the bucket list page, and enter a specified path to enter the bucket and manage its resources.



7. Uploading file/folder

Upload Feature	Description	Directions
Uploading files	COSBrowser allows you to upload a single file or multiple files in batches in different ways.	You can upload files in the following ways. In the specified bucket or path:1. Click Upload Files to upload files directly. 2. Right-click in the blank space of the file list and select Upload Files to upload files. 3. Drag a file to the file list pane.
Uploading a folder and the files contained	If the name of the file/folder to upload already exists in the bucket or path, the old file/folder will be overwritten by default.	You can upload a folder in the following ways. In the specified bucket or path:1. Click Upload Folder to upload a folder directly. 2. Right-click in the blank space of the file list and select Upload Folder to upload a folder. 3. Drag a folder to the file list pane.
Incremental upload	Incremental upload compares the files to upload with existing files in the bucket before the upload. An existing file with the same name in the bucket will be skipped.	You can perform the following two steps to use incremental upload. In the specified bucket or path:1. Upload as you do with a folder and click Next.2. In Storage method, select Skip. Then, click Upload to begin the incremental upload.

Note:

If you need to upload files in batches, you are advised to use a computer with a 4-core CPU and 16 GB RAM, which allows you to upload up to 300,000 files at a time.

8. Downloading file/folder

Download Feature	Description	Directions
Downloading files	COSBrowser allows you to download a single file or multiple files in batches in different ways.	You can download a file in the following ways:1. Select the desired file and click Download in the UI.2. Right-click the file and select Download.3. Drag the file to the local file system.
Downloading a folder and files contained	If the name of the file/folder to download already exists in the local file system, it will be renamed by default.	You can download a folder and the files contained in the following ways:1. Select the desired folder and click Download in the UI.2. Right-click the folder and select Download.3. Drag a folder to the local file system.
Incremental download	Incremental download compares the files to download with local files before the download. An existing file with the same name will be skipped.	You can perform the following steps to use incremental download:1. Right-click the desired file or folder.2. Click Advanced Download and select Skip in the pop-up window.3. Click Download Now to download the incremental unique file/folder.

Note:

If you need to download files in batches, you are advised to use a computer with a 4-core CPU and 16 GB RAM, which allows you to download up to 300,000 files at a time.

9. Deleting file/folder

Select the desired file/folder and click **More > Delete** at the top of the page or right-click the file/folder and select **Delete** to delete it. Batch deletion is supported.

10. Synchronizing file

The file synchronization feature allows you to upload specified files in your local folders to a bucket in real time or at a scheduled time. Detailed directions are as follows:

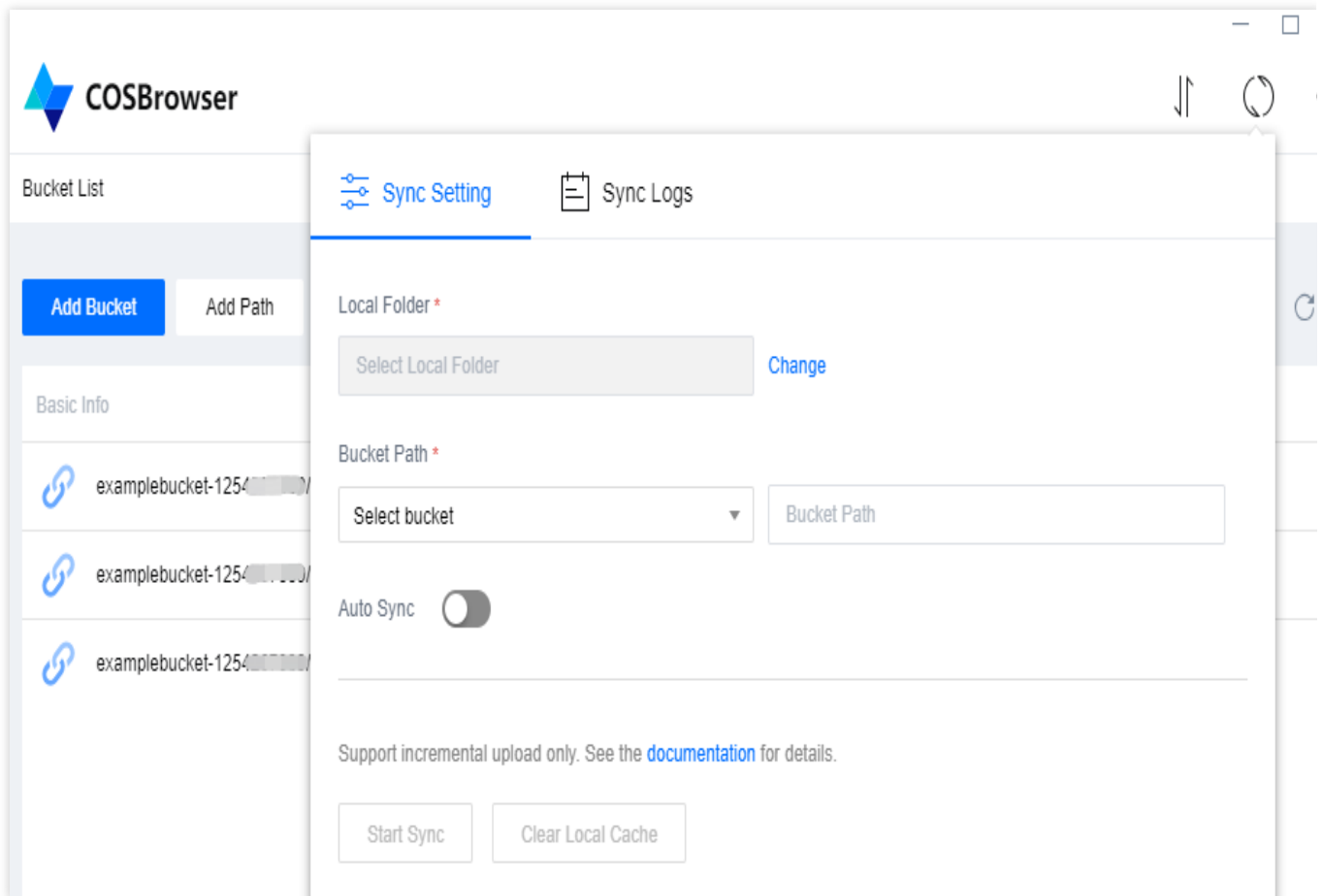
1. Click **Toolbox > File Sync** in the top-right corner of the page.
2. Configure the following information in the pop-up window.

Local Folder: Specify the local folder for file synchronization.

Bucket Path: Specify the destination COS bucket directory.

File Extension Filtering: Specify the filename extensions for filtering. Separate two extensions with a semicolon (;). Files with the specified extensions will be ignored.

Sync Type: You can select one-time, automatic, or scheduled sync and click **Start Sync** to enable file sync.



3. Click **Sync Logs** to view the file sync logs.

Note:

Synchronization means that when the file is uploaded, the system automatically identifies whether the same file exists in the bucket. Only files that do not exist in the bucket are synchronously uploaded.

Currently, only synchronizing local files to the bucket is supported. Reverse operation is not supported.

The file sync feature supports manual, automatic, and scheduled sync.

11. Copying and pasting file

To copy a file/folder, select the desired file/folder in the specified bucket/path and click **Copy** in **More** at the top of the UI. Alternatively, you can right-click it and select **Copy**. After the file/folder is successfully copied, you can paste it to **another bucket or path**. You can copy and paste multiple files/folders in batches.

Note:

If the name of the file/folder to paste already exists in the destination path, the old file/folder will be overwritten by default.

12. Renaming file

Select a file you want to rename, right-click, and select **Rename**. Alternatively, you can click **Rename** in **More Actions** on the right of the file. Then, enter your file name, and click **OK**.

Note:

Folders cannot be renamed.

13. Creating folder

To create a folder in the specified bucket or path, click **Create Folder** in the UI, or right-click in the UI and select **Create Folder**, enter the folder name, and click **OK**.

Note:

The folder name can contain up to 255 characters, including digits, letters, and visible characters.

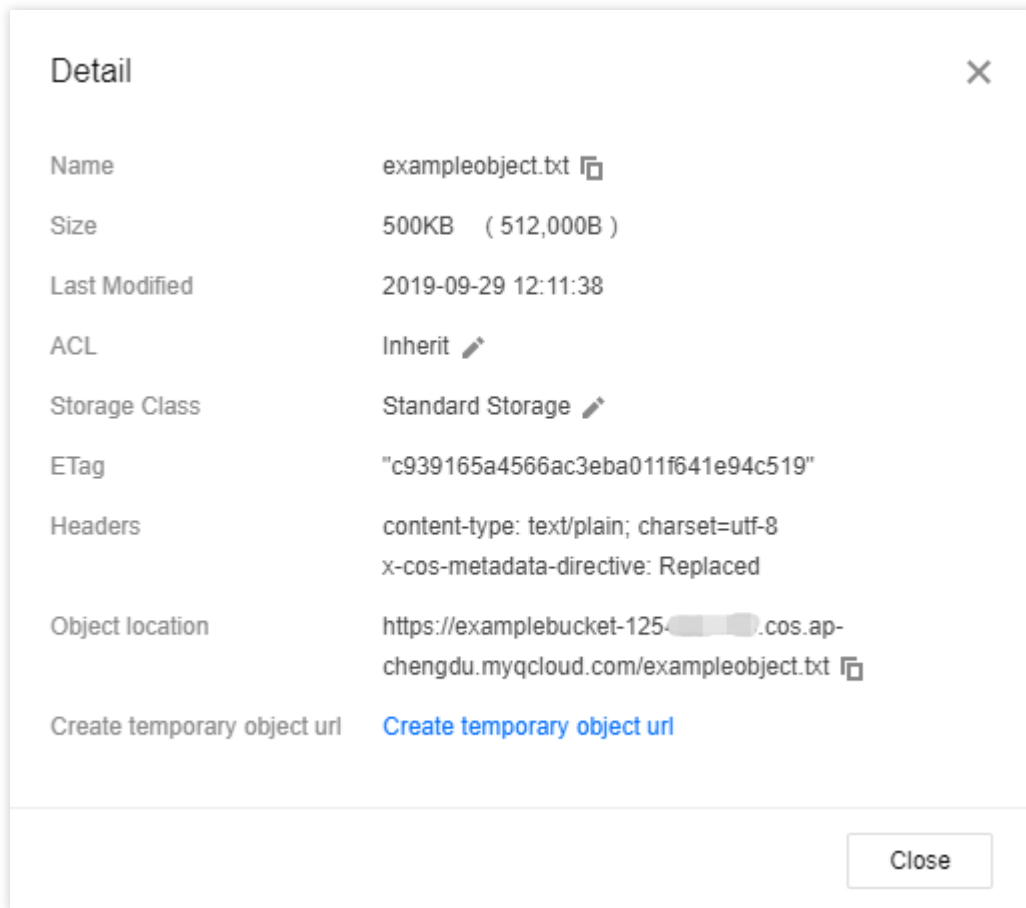
The folder name cannot contain special characters such as `\\ / : * ? " | < > .`

`..` cannot be used as the folder name.

The folder cannot be renamed.

14. Viewing file details

To view the details of a file, tap its filename or right-click it and select **Detail**. File details include filename, file size, modification time, access permission, storage class, ETag, headers, endpoint, object location, and an option to create a temporary object URL.



15. Generating file URL

Each file stored in COS can be accessed through a specific URL. If a file is private-read, you can request a temporary signature to generate a temporary access URL with a certain validity period.

You can generate a file URL in the following ways:

In the table view, click the Share icon on the right of the object to generate a URL and copy it. If the file is public-read, the URL will not carry a signature and be valid permanently. If the file is private-read, the URL will carry a signature and be valid for 2 hours.

Right-click a file and select **Copy Link** to generate a URL and copy it. If the file is public-read, the URL will not carry a signature and be valid permanently. If the file is private-read, the URL will carry a signature and be valid for 2 hours.

On the file details page, click **Create link with expires** to set the temporary URL, URL type, and validity period for the specified endpoint (available only when a CDN acceleration endpoint is enabled).

16. Sharing file/folder

Click **Share** in the operation column or the context menu to share a COS folder. You can also set a validity time for the link.

Note:

You can only share a single folder but not multiple files.

If multiple users share a folder, the file content may be hard to manage. In this case, you are advised to enable versioning for your bucket so that you can roll back to the desired historical version.

Share Folders/Files

Name

doc ... 1 folders/files.

Permissions

☒ Read only ☐ Read & write

Valid time

2

hours

The longest valid time is 2 hours (If log in with a sub-account, the maximum validity period can be extended to 1.5 days)

Extraction code

☒ Auto generate ☐ Custom

ee9b31

Please enter the 6-digit extraction code

OK

Close

Parameter	Description
Permissions	You can set the access permission for the shared folder. Read only: Pulls the folder list and downloads files in the folder from the access URL. Read & write: Pulls the folder list and downloads files in the folder, uploads files to the folder, and creates folders using the access URL.
Valid time	In minutes, hours, or days If you log in to the client using a key, the validity period can be 1 minute to 2 hours for the root account, and 1 minute to 30 days for sub-accounts. If you log in to the client using a Tencent Cloud account, the longest validity period is 2 hours. The default value is the longest validity period allowed for the current user.
Extraction code	A 6-character code automatically generated by the system. You can customize one as needed (numbers, letters, and symbols are supported).

Note:

When the link is valid, users receiving the link and extraction code can access the folder.

17. Exporting file link

COSBrowser allows you to export file links. In the top-right corner of the UI, click the toolbox icon, select **Export File Link** in the toolbox pop-up window, select the bucket, folder path (for example, to export the `folder` folder under the root path, enter `folder/`), and export destination path of the target files, and click **Export**.

18. Previewing file

COSBrowser allows you to preview media files, including images, videos, and audio. To preview a media file, double-click it or right-click it and select **Preview** or **Playback** in the context menu. On the file preview or playback screen, you can click:

Copy Link to generate a file access URL and copy it.

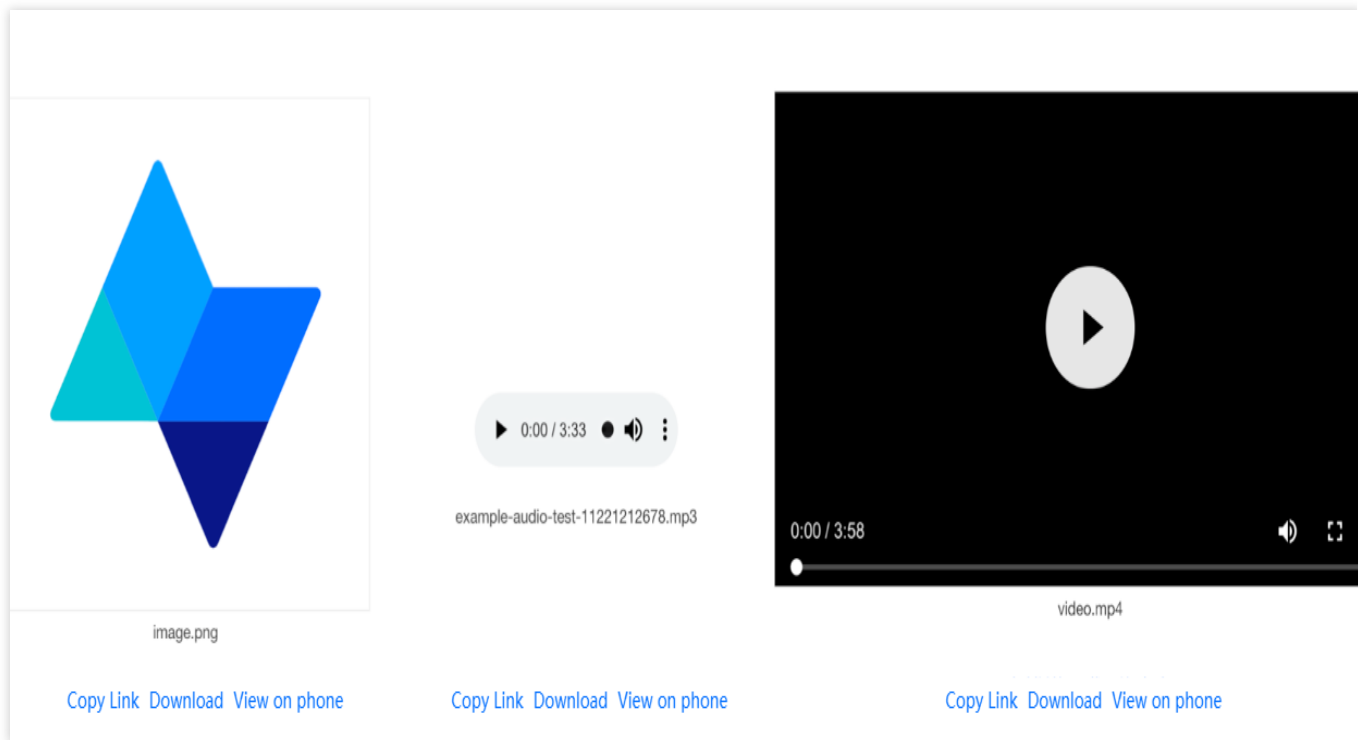
Download to download the file to the local file system. If a file with the same name already exists in the local file system, it will be overwritten by default.

View on phone to generate a QR code for the file, which can be scanned on a mobile phone for direct view.

Note:

Preview is available for images in most formats, .mp4 and .webm videos, and .mp3. and .wav audios.

Please note that file preview will incur downstream traffic.



19. Searching for file

To search for a file, enter the filename in the search box at the top right of the bucket. COSBrowser supports prefix search and fuzzy search.

20. Searching for bucket

To quickly locate a bucket, enter the bucket name in the search box above the bucket list on the left.

21. Viewing historical versions or incomplete multipart uploads

If versioning is enabled for your bucket, you can click **View > Multiversion list** above the file list to view the historical versions. Prefix search and clearing all historical versions (retaining the latest version only) are supported.

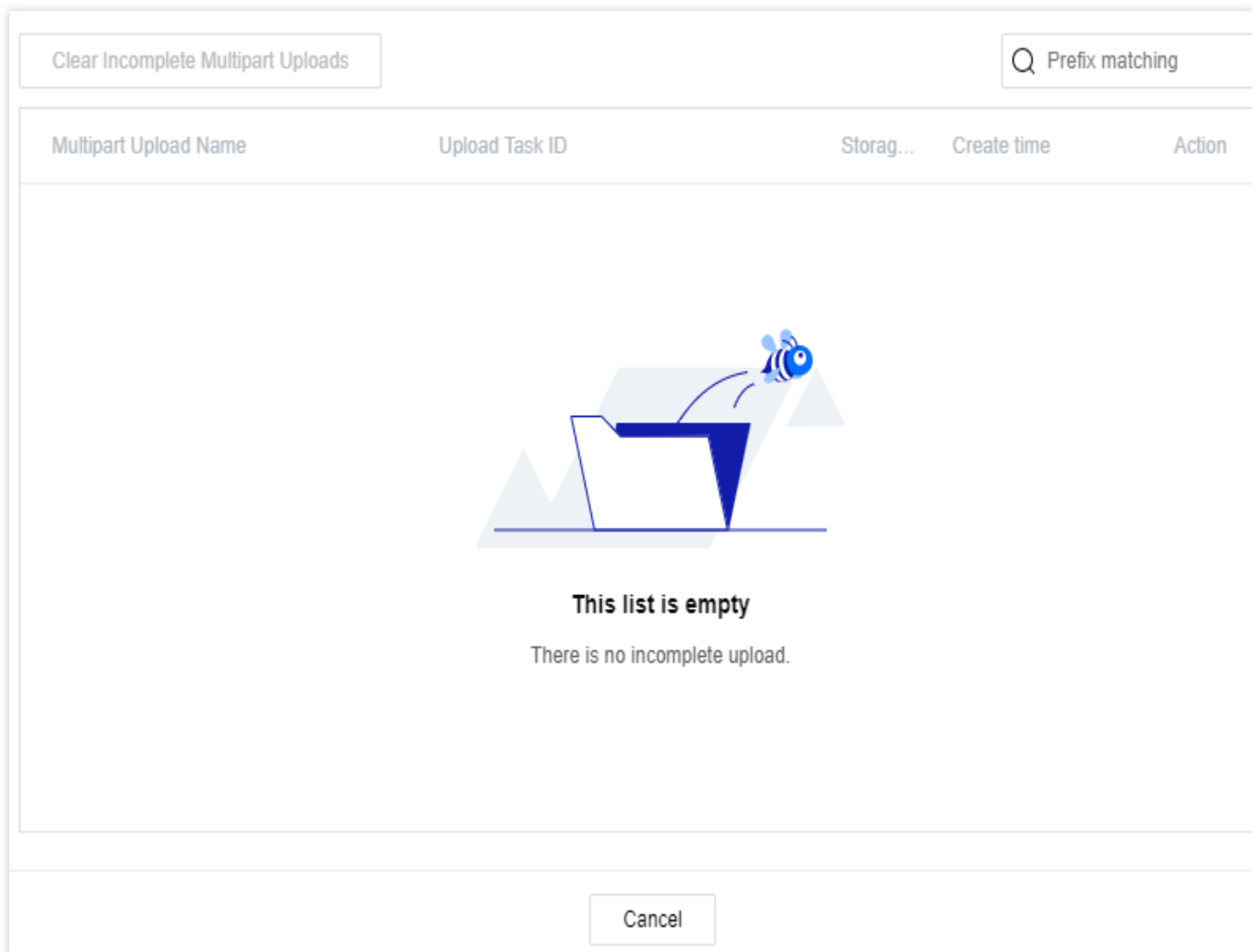
Multiversion list

Q Prefix matching

Name	Size	Last Modified	VersionId	Action
<div><div></div><div>index.html</div></div>	169.71KB	2019-09-19 15:47	-	<div><div></div><div></div></div>
No history files				
<div><div></div><div>video.mp4</div></div>	169.71KB	2019-09-19 17:40	-	<div><div></div><div></div></div>
No history files				
<div><div></div><div>exampleobject.txt</div></div>	500KB	2019-09-29 12:11	-	<div><div></div><div></div></div>
No history files				
<div><div></div><div>image.png</div></div>	9.72KB	2019-11-06 16:01	-	<div><div></div><div></div></div>
No history files				
<div><div></div><div>manifest.csv</div></div>	169.71KB	2019-11-27 18:27	-	<div><div></div><div></div></div>

Cancel

If you pause or cancel an ongoing upload, incomplete multipart uploads may be generated. You can click **View > Incomplete Multipart Uploads** above the file list to view them. Prefix search and clearing all incomplete multipart uploads are supported.



22. Comparing files

In the top-right corner of the UI, click the toolbox icon, select **File Comparison** in the toolbox pop-up window, select the local folder and the bucket (you need to select the region, bucket, and directory) for comparison, and click **Start Comparison**.

23. Transcoding videos

In the top-right corner of the UI, click the toolbox icon, select **Video Transcoding** in the toolbox pop-up window, select a bucket for which the media processing service has been enabled, click **Create Transcoding Job**, select the target media file and transcoding template, enter the output file name and storage path, specify the region, bucket, and directory, and click **Transcode**.

24. Generating authorization code

You can generate authorization codes to temporarily authorize the specified buckets, resources in a bucket, and operations. It is more flexible than folder sharing, as it can grant custom operation permissions for specified directories. You can generate temporary `SecretId` , `SecretKey` , token, and authorization code for other users to log in to the client temporarily.

The operations are as detailed below:

In the top-right corner of the UI, click the toolbox icon, select **Generate Authorization Code** in the toolbox pop-up window, select the bucket and scope of resources for authorization, select the authorized operations such as read/write in the policy permission settings, set the authorization code validity period, and click **OK**.

25. Processing image

The image processing feature of COSBrowser supports basic image processing operations, such as scaling, cropping, rotation, and text and image watermarking. It can also generate processed image links.

Select the target bucket, click the toolbox icon in the top-right corner of the UI, and select **Image Processing** in the toolbox pop-up window. In the image processing pop-up window, select the target image file, configure the feature parameters, and click **Image Preview** to generate the link of the output image.

Settings

System Feature	Description	Directions
Setting up proxy	COSBrowser uses the system-configured proxy to connect to the Internet. Please make sure that your proxy is set up properly or disable the proxy configuration if it fails to connect to the internet.	<ol style="list-style-type: none">1. Select Advanced Setting > Proxy.2. Set up a proxy to connect to the internet.
Setting the maximum number of concurrently uploaded/downloaded files	COSBrowser allows you to set the maximum number of concurrently uploaded and downloaded files.	<ol style="list-style-type: none">1. Select Advanced Setting > Upload (or Download).2. Set the maximum number of concurrently transferred files.
Setting the maximum number of concurrently uploaded/downloaded parts	COSBrowser supports uploading/downloading a file in multiple parts. When the file to be transferred exceeds a certain size, multipart transfer will be performed by default.	<ol style="list-style-type: none">1. Select Advanced Setting > Upload (or Download).2. Set the maximum number of concurrently transferred parts.

Setting the number of retries upon upload/download failure	COSBrowser will retry failed tasks by default when transferring files.	<ol style="list-style-type: none">1. Select Advanced Setting > Upload (or Download).2. Set the number of retries upon transfer failure.
Setting the single-thread upload/download speed limit	COSBrowser supports limiting the upload and download speeds for a single thread. Total upload (download) speed limit = Single-thread upload (download) speed limit x Number of concurrent files x Number of concurrent parts	<ol style="list-style-type: none">1. Select Advanced Setting > Upload (or Download).2. Set the single-thread upload (or download) speed limit in MB/s.
Setting upload check	COSBrowser supports checking files online after upload to verify whether their size and status are correct.	<ol style="list-style-type: none">1. Select Advanced Setting > Upload.2. Select Secondary verification after uploading.
Viewing local log	COSBrowser will record all the performed operations in the `cosbrowser.log` local file.	<ol style="list-style-type: none">1. Select Advanced Setting > About.2. Click Local Logs to enter the local log directory.

User Guide for Mobile Version

Installation and Login

Last updated : 2024-03-25 15:05:23

Download and Installation

Downloading software

OS	System Requirements	Download Address
Android	Android 4.4 or Later	Android
iOS	iOS 11 and later	iOS

Installation

COSBrowser Mobile Version is currently available in most app platforms such as MyApp and App Store. You can download it from the above download address or in an app platform.

Login Options

COSBrowser Mobile Version supports the following login options:

Login with email: If your Tencent Cloud account was created through email or associated with a specific email address, you can log in to COSBrowser by entering the email address and password.

Login with permanent key: You can log in using your TencentCloud API key (SecretId and SecretKey; project key is not supported), which can be created or obtained on the [API Key Management](#) page in the CAM console. After successful login, the account will be kept logged in permanently.

Note:

Sub-accounts can log in with a key.

Mobile Version Features

Last updated : 2024-01-06 16:15:34

COSBrowser mobile app allows you to easily view and manage COS resources anytime you want. Supported operations are as described below.

Data Monitoring

Operation	Description
Usage overview	You can view the recent data usage.
Bucket monitoring	You can view the recent data usage by bucket.
Widget	You can view the data usage without opening the app.

Bucket Management and Operations

Operation	Description
Viewing bucket list	You can view buckets by region.
Adding access path	If you log in with a sub-account that does not have permission to access the bucket list, you can initiate an access request via Add Access Path .
Bucket creation	You can create buckets.
Bucket search	You can fuzzy search for buckets by keyword in the bucket list.
Viewing bucket's basic information	You can view the basic information of buckets, such as name, region, and creation time on Mobile Version.
Bucket permission management	You can modify the public and user permissions of buckets.
Enabling global acceleration	You can enable the global acceleration feature for buckets.
Bucket transfer configuration	You can set the domain names for uploads and downloads.

Object Management and Operations

Operation	Description
Folder creation	You can create folders in buckets.
Folder deletion	You can delete all files in the current directory and all sub-directories.
File upload	You can use COSBrowser Mobile Version to upload local and remote files as well as files from other apps and file managers to COS. You can also set file information such as storage class, access permissions, encryption method, object tags, and metadata during upload.
File backup	COSBrowser provides the automatic backup feature. After you enable backup, it will automatically back up files in your album to the specified bucket.
File download	You can download files in buckets to the app or save them to the local album.
Batch operation	You can batch upload, download, delete, copy, and move files in buckets.
File preview	You can preview images, videos, audios, documents, and files in many different formats in buckets.
Online file decompression	You can decompress .zip, .tar, and .gz packages online.
File sharing	COSBrowser provides the folder and file sharing feature for you to quickly collect or share data in buckets to other users.
File renaming	You can rename files in buckets.
File search	You can fuzzy search for files by file type, such as folder, image, video, document, and audio.
Object sorting or filtering	You can sort files in buckets by filename, size, and modification time and filter them by storage class.
File permission management	You can set file access permissions, which have a higher priority than those for buckets.

Bucket Management and Operations

Last updated : 2024-01-06 16:15:35

Note:

COSBrowser for iOS v2.7.6 is used as an example here. For other versions, see [Changelog](#).

Viewing Bucket List

COSBrowser Mobile Version displays bucket lists by region. You can view buckets by region and click **Resources** at the bottom to view created buckets.

Adding an Access Path

If you log in with a sub-account that has no access to the bucket list, you can add the specified path to a bucket or directory to manage resources by tapping **Add Access Path** in the upper-right corner of the bucket list page.

Note:

This feature is supported only for sub-accounts.

Directions

1. Go to the bucket list page and click **+** in the top-right corner.
2. In the pop-up operation list, click **Add Access Path** and enter an access path authorized by the root account.

Creating a Bucket

You can create a bucket by specifying the bucket name, region, and access permissions on COSBrowser Mobile Version.

Directions

1. In the bucket list, tap **+** in the top-right corner.
2. In the pop-up operation list, tap **Create Bucket**.
3. On the bucket creation page, configure the following information:

Name: A custom bucket name, which cannot be modified once configured. For more information, see [Bucket Overview](#).

Region: a COS region corresponding to the physical location where your business or users are distributed. This parameter cannot be modified once configured. For more information about regions, please see [Regions and Access Endpoints](#).

Access Permissions: Three access permissions are available for buckets by default: "Private Read/Write", "Public Read/Private Write", and "Public Read/Write". You can modify it if needed. For more information, please see [Setting Access Permission](#).

4. After confirming that everything is correct, tap **OK**.

On the bucket list page, you can view the newly created bucket.

Searching for Bucket

If you have many buckets, you can enter a bucket name in the search box at the top of the bucket list page to fuzzy search for buckets.

Viewing Bucket's Basic Information

1. On the bucket list page, tap ... on the right of the target bucket.
2. In the pop-up operation list, tap **Details** to view the bucket's basic information.

Basic Information includes bucket name, region, creation time, and MAZ status.

Managing Bucket Permission

You can use the COS app to set or modify bucket access permissions of the following two types.

Public permissions: include private read/write, public read/private write and public read/write. For more information, see **Types of Permission** under [Bucket Overview](#).

User ACLs: the root account has all bucket permissions (full control) by default. You can add sub-accounts and grant them permissions including read/write, read/write ACL, and even **full control**.

1. Go to the bucket list page and tap ... on the right of the target bucket.
2. In the pop-up operation list, tap **Permission** to enter the bucket permission page.

Modifying public permission

On the bucket permission page, tap a public permission configuration item to modify it.

Setting user permissions

On the bucket permission page, tap **Add User** to set bucket access permissions.

Editing or deleting bucket permission

Select the target user permission, swipe left, and click the displayed **Edit** or **Delete** button to edit or delete the user permission.

Enabling Global Acceleration

You can enable the global acceleration feature for your bucket on COSBrowser Mobile Version. It allows users across the world to quickly access the bucket and improves your business access success rate and stability. It can accelerate both uploads and downloads.

Note:

Enabling global acceleration will not affect the existing default bucket domain name. You can still use them.

Directions

1. Go to the bucket list page and tap ... on the right of the target bucket.
2. In the pop-up operation list, tap **Transfer** to enter the transfer list page.
3. Tap the global acceleration configuration item to enable or disable it as needed.

Configuring Bucket Transfer

You can enable data transfer via a custom domain name for a bucket on the **Transfer** list page.

Setting upload domain name

After global acceleration is enabled for a bucket, you can specify the global acceleration endpoint for file upload to a bucket, which will be used first for file uploads once configured.

Setting download domain name

After a custom endpoint is set for a bucket, you can specify it for file download from a bucket, which will be used first for file downloads once configured.

File Management and Operations

Last updated : 2024-01-06 16:15:35

Note:

COSBrowser for iOS v2.7.6 is used as an example here. For other versions, see [Changelog](#).

Creating Folders

COSBrowser Mobile Version allows you to create a folder in a bucket as follows:

1. On the bucket's file list page, click **+** in the top-right corner.
2. In the displayed operation list, click **Create Folder**.
3. On the **Create Folder** page, enter a folder name and click **OK**.

Deleting Folders

Note:

Deleting a folder will delete all files under it and its sub-directories.

Directions

1. Click **...** on the right of the target folder.
2. In the displayed operation list, click **Delete**.

Uploading Files

You can use COSBrowser Mobile Version to upload local and remote files and files from other apps or the file manager to COS and set information such as file storage class, access permission, encryption method, object tags, and metadata during upload.

Uploading image/video

COSBrowser allows you to batch upload images or videos from your album to COS.

Directions

1. On the bucket list or file list page, click **+** in the top-right corner to display the operation list.
2. In the operation list, click **Upload Images**.
3. In the file list of the displayed album, select the target files and click **Next**.
4. (Optional) Configure the upload parameters. COSBrowser allows you to set the following file attributes during upload:

Storage Class: Select the storage class for your object as needed. This field is set to `STANDARD` by default. For more information, see [Overview](#).

Note:

If your bucket has MAZ configuration enabled, you can only select a MAZ-enabled storage class, such as MAZ_STANDARD. If it also has INTELLIGENT TIERING configuration enabled, you can also select MAZ_INTELLIGENT TIERING.

Access Permissions: select the access permission for your object as needed. This field is set to `Inherit` by default (inheriting permissions of the bucket). For more information, please see [Basic Concepts of Access Control](#).

Server-Side Encryption: configure server-side encryption for the object you want to upload. COS will automatically encrypt your data as it is written and decrypt it when you access it. Currently, COS offers two encryption types: SSE-KMS (only available in Beijing, Shanghai, and Guangzhou regions) and SSE-COS. For more information, please see [Server-side Encryption Overview](#).

-Object Tag: An object tag is composed of a tag key, equal sign (=), and a tag value, for example, `group = IT`. You can set, query, and delete tags of a specified object.

Metadata: object metadata, or HTTP header, is a string sent by the server over HTTP before it sends HTML data to the browser. By modifying HTTP headers, you can modify how the webpage responds as well as certain configurations, such as caching time. Modifying an object's HTTP headers does not modify the object itself. For more information, please see [Custom Headers](#).

5. Click **Upload**.

Uploading file through link

COSBrowser allows you to upload a file through the file link. Every time you enter the app, it will check the current clipboard. If there is any valid file link on the clipboard, a message will pop up asking you whether to upload the file through the link. In this case, you simply need to click **Upload Now**.

You can also upload a file through the link as follows:

1. On the bucket list or file list page, click **+** in the top-right corner to display the operation list.
2. Click **Upload Link**, paste the link for file upload into the text box, select an upload path, and click **Upload**.

Uploading file shared by third-party app

You can also share a file from another app to COSBrowser to upload it.

Note:

This feature requires that the third-party app supports file sharing to other apps.

Directions

Take QQ as an example:

1. In QQ, click a file to preview it, click ... in the top-right corner, and select **Other Apps**.
2. Click COSBrowser in the app list.

Uploading file from file manager

COSBrowser can upload files from the system file manager ("Files" on iOS and the corresponding system file manager app on Android) to COS.

Directions

1. On the bucket list or file list page, click **+**.
2. In the displayed operation list, click **Upload Files**.
3. On the displayed file manager page, click the target file.

Backing up Files

COSBrowser provides the auto backup feature. After this feature is enabled, COSBrowser will automatically back up the files in your album to the specified bucket. To help you manage backup files, COSBrowser displays the backup data as a separate module on the homepage.

Note:

The album backup feature is only supported for the root account.

The album module displays only image and video files. To view all files, go to the bucket list and search for the backup bucket.

Setting backup

Go to **Personal > Album Backup** and toggle on **Automatic Photo Backup** or **Automatic Video Backup**.

Automatic Photo Backup: After it is enabled, all images in the album will be backed up.

Automatic Video Backup: After it is enabled, all videos in the album will be backed up.

Back up over Wi-Fi Only: After it is enabled, files will be backed up only over a Wi-Fi network.

Region: After the backup region is selected, a bucket named `from-phone-date-APPID` will be created in the region by default.

Note:

The region can be set only before backup is enabled and cannot be modified after the settings are saved.

Enable Smart Storage: This option is suitable for buckets with INTELLIGENT TIERING enabled. After it is enabled, files in the album will be uploaded to COS in INTELLIGENT TIERING. COS will automatically switch between the STANDARD and STANDARD_IA classes based on the access frequency of INTELLIGENT TIERING objects with no data retrieval fees incurred, which reduces your storage costs. For more information, see [INTELLIGENT TIERING Overview](#).

Managing backup file

COSBrowser supports batch file upload and download.

Directions

1. On the **Album** page, click the batch operation icon in the top-right corner to display operation buttons.
2. Select the target files and click **Download** or **Delete** to download or delete the files.

Adding file to backup bucket

You can also add image or video files from other buckets to the backup bucket to quickly preview them in the **Album** module.

Directions

1. Click ... on the right of the target image or video file.
2. In the displayed operation list, click **Add to Album**.
3. Enter the **Album** module again to view the file.

Downloading Files

COSBrowser allows you to download files from COS to your local file system. It also enables you to save images to the system album.

Downloading to app

COSBrowser provides multiple download entries for you to download files anytime, anywhere.

Method 1:

- 1.1 On the file list page, click ... on the right of the target file.

1.2 In the displayed operation list, click **Download**.

Method 2:

On the **File Details** page, click In the displayed operation list, click **Download**.

Method 3:

On the file preview page, click ... in the top-right corner. In the displayed operation list, click **Download**.

n>

Saving to album

COSBrowser allows you to save images to the local album as follows:

1. On the file list page, click ... on the right of the target file to display the operation list.
2. Click **Download** and select **Save to Album**.

Batch Operations

COSBrowser allows you to batch download, delete, copy, and move files from a bucket.

Batch download

1. Click a bucket to enter the file list page and click ... in the top-right corner.
2. In the displayed operation list, click **Batch Operation**.
3. Batch select files and click **Download** on the bottom operation bar.

You can also click the **Transfer** button in the top-right corner of the file list page to view the jobs on the transfer list page.

Deleting domain names by batches

1. Click a bucket to enter the file list page and click ... in the top-right corner.
2. In the displayed operation list, click **Batch Operation**.
3. Batch select files and click **Delete** on the bottom operation bar.

Batch copy

Note:

When you use COSBrowser to copy a source file, its information such as ACLs, policies, and tags will also be copied.

1. Click a bucket to enter the file list page and click ... in the top-right corner.
2. In the displayed operation list, click **Batch Operation**.

3. Batch select files and click **More > Copy** on the bottom operation bar.

Batch moving

Note:

When you use COSBrowser to move a source file, its information such as ACLs, policies, and tags will also be moved.

1. Click a bucket to enter the file list page and click ... in the top-right corner.
2. In the displayed operation list, click **Batch Operation**.
3. Batch select files and click **More > Move** on the bottom operation bar.

Previewing Files

COSBrowser allows you to preview various file formats such as image, audio, video, and document. It also supports online file decompression.

Online image preview

COSBrowser allows you to preview images online without downloading them.

Enabling image preview

You can preview images in the following three methods:

1. Enable the feature globally

Select **Personal > Settings** and toggle on **Image Preview**.

2. Enable the feature temporarily

When you launch the app for the first time and enter the file list page, the app will check whether the current list page contains images, and if yes, a pop-up window will appear asking you whether to enable image preview. You can click **Enable Preview** to quickly enable the **image preview** feature. If you don't need this feature, you can click **Cancel**. If you want to use it in the future, you can manually enable it in **Personal > Settings**.

3. Enable the feature for one image

If you have disabled image preview in **Personal > Settings**, and it's not the first time that you open the app, when you enter the image details page, the app will ask you whether to enable preview. In this case, you can click **Click to Preview** to preview a file.

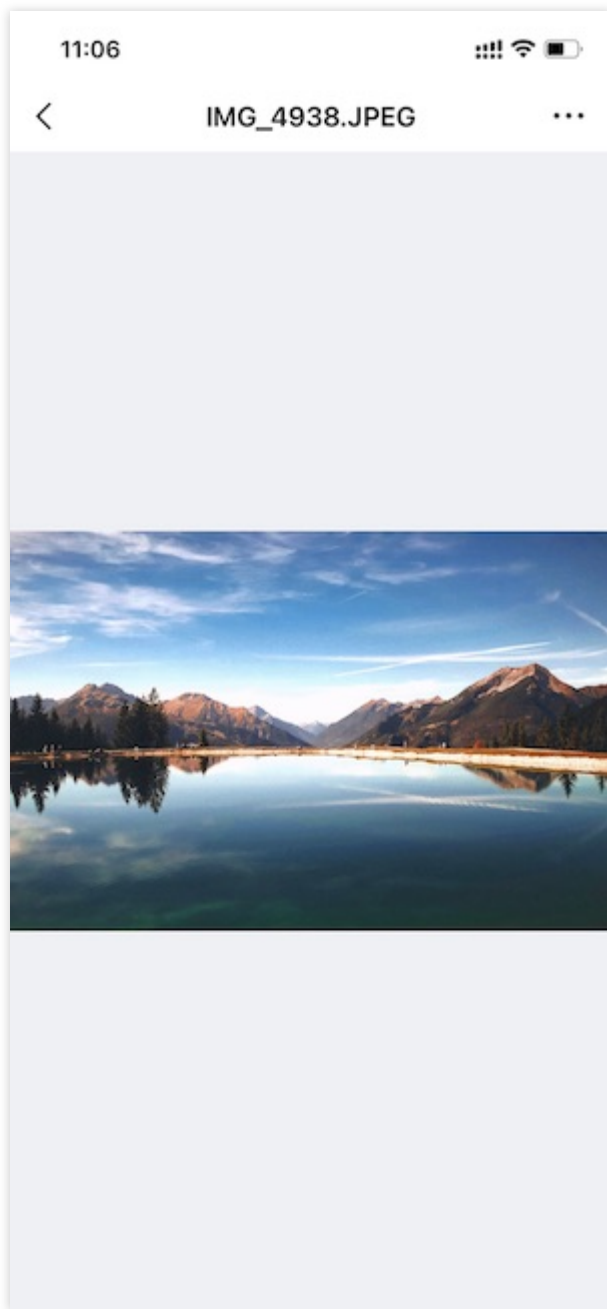
Note:

This switch doesn't change the status of the global **image preview** toggle.

Big image mode

COSBrowser provides the big image mode feature for you to view image details more clearly.

1. On the file list page, click the target image to enter the **File Details** page.
2. Click the image to enter the big image mode, in which you can zoom in the image with two fingers to view its details.



Playing back audio

COSBrowser can play back audio files in MP3, OGG, AAC, WMA, WAV, APE, or FLAG format.

Directions

1. On the file list page, click an audio file to enter the **File Details** page.

2. Click the playback button.

You can speed up or pause the playback.

Playing back video

COSBrowser provides a simple online video playback feature, which supports various formats such as AVI, WMV, MPEG, RM, RMVB, MKV, MOV, QT, and MP4. You can speed up or pause the video playback during watch. You can also play back the video in a floating window after the app enters the background.

Directions

1. On the file list page, click the target video file to enter the **File Details** page.
2. Click the playback button.

You can speed up or pause the playback.

Note:

After the app enters the background, you can still watch the video in a floating window.

Previewing document

COSBrowser allows you to preview files in multiple formats such as PDF.

Directions

1. Click the target document to enter the **File Details** page. If you haven't enabled the document preview feature, click **Click to Enable**.
2. On the document preview feature configuration page, click **Enable Now**.
3. Return to the previous page and click **Click to Preview**.
4. On the document preview page, swipe left to view the next page.

Decompressing Files

COSBrowser allows you to decompress files in ZIP, TAR, or GZ format online. The extracted files will be stored in the current directory.

1. On the file list page, click ... on the right of a compressed file.
2. In the displayed operation list, click **Online Decompression**.
3. After the file is successfully decompressed, the extracted files will be displayed in the current directory.

Sharing Files

COSBrowser provides the folder and file sharing feature for you to quickly collect data or share the data in a bucket with other users.

Sharing folders

Note:

You can only share a single folder but not multiple files.

A sharing duration of up to 2 hours and up to 1.5 days can be set for the root account and a sub-account respectively.

If multiple users share a folder, the file content may be hard to manage. In this case, we recommend you enable versioning for your bucket so that you can roll back to the desired historical version.

Generating sharing QR code/link

The detailed steps are as follows:

1. Click ... on the right of a folder and click **Share** in the displayed operation list.
2. On the displayed sharing page, you can select QR code or link for sharing.
3. (Optional) Configure the sharing parameters, which are as detailed below. You can skip this step to keep the parameters as default.

Permission: Sets the access permission for the shared folder.

Can view: Pulls the folder list and downloads files in the folder using the access URL.

Can edit: Pulls the folder list and downloads files in the folder, upload files to the folder, and create folders using the access URL.

Validity Period: It is in minutes, hours, or days. The validity period for the root account and a sub-account is two hours and 24 hours respectively by default, which cannot be changed.

Password: A 6-character password automatically generated by the system. You can customize one as needed (numbers, letters, and symbols are supported).

4. Click **Generate Link** or **QR Code Sharing** to generate the sharing link or QR code.

Viewing folder shared by another user

You can open a folder shared by another user from a mobile or desktop client.

Method 1. View on mobile device

1. On the **Login** or **Personal** page, click **Scan** to scan the QR code.
2. Enter the password and click **OK** to view the shared folder.

Method 2. View on PC

1. Click **Log in with Shared Link** on the **Login** page.
2. Enter the obtained URL address and password and log in.

Method 3. View in browser

1. Open the browser and enter the shared URL to open it.
2. Enter the password and click **Extract** to enter the shared folder.

Sharing files

Each file stored in COS can be accessed through a specific URL. You can generate a file URL with the specified domain name if you have set other domain names (such as CDN acceleration domain name and custom origin domain name) for the bucket. If a file is private-read, you can request a temporary signature to generate a temporary access URL with a certain validity period.

Note:

If you log in with a temporary key, you cannot configure the validity period of the file link, which is 1 hour by default. If the file is public-read, the URL will not carry a signature and will be valid permanently. If the file is private-read, the URL will carry a temporary signature and will be valid for one hour.

Directions

1. Click a file to enter the **File Details** page.
The configuration items are described as follows:
Specify Domain: Set the domain name for the URL (optional).
Validity Period: Set the validity period for the URL (optional).
2. After confirming that the configuration is correct, click **Generate Link**.
3. Send the generated file URL.

Renaming Files

Note:

Folders cannot be renamed.

Directions

1. Click ... on the right of a file.
 2. In the file operation list, click **Rename**.
 3. In the pop-up window, enter the new filename and click **OK**.
- If you select **Overwrite Duplicate File**, the original file will be overwritten; otherwise, a new file will be generated. You can also enable overwriting globally in **Personal > Settings > Default Upload Options > Rename Duplicate File**.

Searching for Files

COSBrowser Mobile Version provides the fuzzy search and search by type features. You can use a prefix to search for files with that filename prefix in the current folder and its sub-folders. You can also specify the file type first to search for files in that type.

Search by keyword

You can enter a search keyword to filter out all filenames containing the keyword in the current folder and all its sub-folders.

Search by type

COSBrowser allows you to search for files by type, such as video, folder, audio, document, image, and others.

Directions

Taking folder search as an example:

1. Click the search box and then **Folder**.
2. All files (which are folders here) in the folder type in the current directory will be listed.

Sorting or Filtering Objects

COSBrowser allows you to sort and filter files in a bucket.

Note:

Currently, sorting by file name/file size/modification time, and filtering by storage class are supported.

Directions

1. Enter the file list page and click the filter and sort button on the right of the search box.
2. In the operation list, sort or filter files in the bucket.

Managing File Permissions

COSBrowser allows you to set file access permissions, which takes priority over that for buckets.

Note:

Object access permissions take effect only when the access is made via the default endpoint. For any access made via a CDN acceleration endpoint or a custom endpoint, bucket access permissions will take effect.

There are limits on the number of ACL rules. For more information, see [Specifications and Limits](#).

Modifying public permission

1. Click the target file to enter the **File Details** page.
2. Click **Permission Info** at the top to enter the permission list page.
3. Click **Public Permission** to modify the access permissions of the file.

Setting user permissions

1. Click the target file to enter the **File Details** page.
2. Click **Permission Info** at the top to enter the permission list page.
3. Click **Public Permission** to modify the user permissions of the file.

You can click **Add User** to add a user permission. Then, you can swipe left on a user permission to edit or delete it.

Data Monitoring

Last updated : 2024-11-18 14:34:00

The data overview feature of COSBrowser Mobile Version displays your COS data usage in the past 30 days, including total traffic, total read and write requests, storage trend, traffic trends (for public network downstream traffic, private network traffic, and CDN origin-pull traffic), trend in the number of requests, trend in the ratio of effective requests, amount of STANDARD_IA data retrieved, and amount of archive data retrieved.

Note:

Current, you cannot view data with a sub-account on Mobile Version. You can log in to a sub-account to view data in the console as instructed in [Viewing Statistics](#).

The data in this feature isn't real-time but has a delay of about two hours, so it is for reference only. You can view more accurate billing and usage data in the console as instructed in [Viewing Statistics](#).

Usage Overview

COSBrowser provides the storage data overview page, where you can view the number of objects, storage usage, number of requests, and traffic.

Bucket Monitoring

COSBrowser provides data overview by bucket. You can view the bucket monitoring data in the following two ways:

1. Tap **Home** at the bottom and tap **User Overview** to switch between buckets to view their monitoring data.
2. Tap **Resources** at the bottom and tap **More** on the right of the target bucket. In the pop-up operation list, tap **Monitoring** to enter the bucket page.

Widget

COSBrowser can be added to the Home Screen as a widget, so you can view the monitoring data anywhere, anytime with no need to open COSBrowser.

Adding widget

1. Download and install COSBrowser for iOS on your iPhone. Then, touch and hold an empty area on the Home Screen until apps jiggle and tap **+** in the top-left corner.
2. Find and tap COSBrowser.
3. Select a layout and tap **Add Widget** at the bottom.

Customizing displayed data

The widget allows you to customize the scope of data to be displayed. You can specify a certain bucket or storage class for tracking and monitoring. The following storage classes can be displayed: STANDARD, STANDARD_IA, and ARCHIVE.

The detailed steps are as follows:

1. Open COSBrowser for iOS and tap **Me > Settings > Widget Configuration** to enter the widget settings page.
2. Set the configuration items as follows:

Select Bucket: If no bucket is selected, the overview data of all buckets under the current account will be displayed. If a bucket is selected, its overview data will be displayed.

Select Storage Class: The widget displays the monitoring data of the STANDARD storage class by default. You can manually select another storage class such as STANDARD_IA and ARCHIVE.

Note:

If you want to quickly reset the configured bucket and storage class, simply tap **Reset**. Then, the monitoring data of all buckets in the STANDARD storage class under the current account will be displayed.

Removing widget

Touch and hold the widget and tap **Remove Widget**.

COSCLI (Beta)

COSCLI Overview

Last updated : 2025-05-09 18:27:38

COS provides the command-line client COSCLI to allow you to upload, download, delete, and perform other operations on COS objects by using simple commands.

COSCLI is written in Go and adopts the Cobra framework. It supports multi-bucket configurations and cross-bucket operations. You can run `./coscli [command] --help` to see how to use COSCLI.

Features

[Generating and Modifying Configuration Files - config](#)

[Creating Buckets - mb](#)

[Deleting Buckets - rb](#)

[Tagging Bucket - bucket-tagging](#)

[Querying Bucket/Object List - ls](#)

[Obtaining Statistics on Different Types of Objects - du](#)

[Uploading/Downloading/Copying Objects - cp](#)

[Syncing Upload/Download/Copy - sync](#)

[Deleting Objects - rm](#)

[Obtaining Object Hash - hash](#)

[Listing Incomplete Multipart Uploads Generated Upon Upload - lsparts](#)

[Aborting Incomplete Multipart Uploads - abort](#)

[Restoring Archive Objects - restore](#)

[Getting Pre-Signed URLs - signurl](#)

[Creating/Obtaining a Symbolic Link - symlink](#)

[Viewing Contents of an Object - cat](#)

[Listing Contents and Statistics Under a Directory - lsdu](#)

[Bucket Versioning - bucket-versioning](#)

Download Address

COSCLI tool supports Windows, MacOS, and Linux operating systems. For details, see [Download and Installation Configuration](#).

Download and Installation Configuration

Last updated : 2025-05-09 18:27:38

COSCLI provides binary packages for Windows, macOS, and Linux, which can be used after simple installation and configuration.

Step 1: Download the COSCLI tool

You can choose the download address for the COSCLI tool based on your business scenario. If your server is in China, it's recommended to use the Domestic Site Download Address (The tool versions of the links here are all the latest versions. If you need to use an earlier version, you can visit [release](#) to retrieve historical versions).

Domestic Site Download Address	GitHub Download Address (for international sites)	SHA256 Checksum
Windows-386	Windows-386	8d415550d4625e0df0b39164326cedd5512d3271c11bebea0dcf9cfc01f6fe46
Windows-amd64	Windows-amd64	a81488cecae7767904e2a111cd8f6d2def0f95f8aba04bb7bcef47b1327bc210
MacOS-amd64	MacOS-amd64	c9c4bb2a021423e3a3ee014c99c8287d484d9a55451f7f75c87d3aaa9417e80f
MacOS-arm64	MacOS-arm64	9253e043985b51219112400a37b39e618ab5c358cf5777193829aa7b9d9ff5f6
Linux-386	Linux-386	e25a856141140c0da262509fa146a424d19f23830cbd4d92e231f7e38d07d8e7
Linux-amd64	Linux-amd64	8a137821c9869d9c07be84e8c2ed69177d9029c1c504e04662e048631112553c
Linux-arm	Linux-arm	07c03d5fc78ef5c714e0e154b8a79fc5850382527e03d7e60b46b26cf9478d59
Linux-arm64	Linux-arm64	34b4d1a82dc5aa23d80c808289d97a03bf101b31871c0a53100de2af2fd25b0e

You can also use the command line to get the COSCLI tool files for MacOS and Linux environments from a domestic site:

MacOS-amd64: `wget https://cosbrowser.cloud.tencent.com/software/coscli/coscli-darwin-amd64`

MacOS-arm64: `wget https://cosbrowser.cloud.tencent.com/software/coscli/coscli-darwin-arm64`

Linux-386: `wget https://cosbrowser.cloud.tencent.com/software/coscli/coscli-linux-386`

Linux-amd64: `wget https://cosbrowser.cloud.tencent.com/software/coscli/coscli-linux-amd64`

Linux-arm: `wget https://cosbrowser.cloud.tencent.com/software/coscli/coscli-linux-arm`

Linux-arm64: `wget https://cosbrowser.cloud.tencent.com/software/coscli/coscli-linux-arm64`

Note:

The current version on GitHub is v1.0.6. To get the latest version, historical versions, and change logs of the tool, please go to [release](#) to view.

Step 2: Install the COSCLI Tool

Windows

1. Take `windows-amd64` version as an example, move the downloaded Windows version of the COSCLI tool to the `C:\Users\< username >` directory.
2. Rename `coscli-windows.exe` to `coscli.exe`.
3. Press `win+r` to open the `Run` program.
4. In the dialog box, enter `cmd` and press `Enter` to open the command line window.
5. In the command prompt, enter the following command.

```
coscli --version
```

If the output is `coscli version v1.0.6`, the installation is successful.

Note:

Under the `Windows` system, the method of using COSCLI in different command line clients may vary slightly. If entering `coscli [command]` does not work properly with COSCLI, please try the format `./coscli [command]`.

MacOS

1. Take `MacOS-amd64` version as an example, run the following command to rename the macOS version COSCLI file.

```
mv coscli-darwin-amd64 coscli
```

2. Run the following command to modify the file execution permission.

```
chmod 755 coscli
```

3. In the command prompt, enter the following command.

```
./coscli --version
```

If the output is `coscli version v1.0.6` , the installation is successful.

Note:

When using COSCLI on a macOS, if the prompt `Unable to open "coscli" because the developer cannot be verified` appears, you can go to `Settings > Security and Privacy > General` and select `Still want to open coscli` , afterwards COSCLI can be used normally.

Linux

1. Take `Linux-amd64` version as an example, run the following command to rename the Linux version COSCLI file.

```
mv coscli-linux-amd64 coscli
```

2. Run the following command to modify the file execution permission.

```
chmod 755 coscli
```

3. In the command prompt, enter the following command.

```
./coscli --version
```

If the output is `coscli version v1.0.6` , the installation is successful.

Step 3: Configure COSCLI Tool

Note:

It is recommended that users use the tool with a [Temporary Key](#) to further enhance its security through temporary authorization. When applying for a Temporary Key, please follow the [Principle of Least Privilege](#) to prevent leaking resources outside the targeted buckets or objects.

If you must use a permanent key, it is recommended to follow the [Principle of Least Privilege](#) to limit the permission scope of the permanent key.

When using COSCLI for the first time, you need to initialize the configuration file, which contains the following two parts:

To authorize COSCLI to access your Tencent Cloud account, you need to configure the secret ID, secret key, and temporary key token.

To add an alias for a common bucket, you need to configure its name, geographical information, and alias. After configuring this information, users can use the alias for bucket operations without having to fill in the bucket name and geographical information again. Adding configurations for multiple common buckets also facilitates cross-bucket or cross-domain operations more easily. If you don't need to configure common bucket information, you can press `Enter` to skip.

When using COSCLI for the first time, it will automatically call `./coscli config init` to create a configuration file at `~/.cos.yaml`. You can complete the setup interactively through the command line. Later, you can also use `./coscli config init` to interactively generate a configuration file in another location for COSCLI. You can use `./coscli config show` to view the location and configuration parameters of the file.

The configuration items in the configuration file are as described below:

Configuration Item	Description
Secret ID	Key ID, it is recommended to use a sub-account key and follow the principle of least privilege to reduce risks. Information on obtaining a sub-account key can be found in Sub-account Access Key Management .
Secret Key	Key, it is recommended to use a sub-account key and follow the principle of least privilege to reduce risks. Information on obtaining a sub-account key can be found in Sub-account Access Key Management .
Session Token	Temporary key token, needs to be configured when using a temporary key; otherwise you can skip by pressing <code>Enter</code> . For more information on temporary keys, see Accessing COS Using a Temporary Key .
Mode	Set the identity mode, supporting enumerated values <code>SecretKey</code> and <code>CvmRole</code> . It can be null, with the default null value being <code>SecretKey</code> , indicating the use of a key request COS. When Mode is <code>CvmRole</code> , it means using Managing Instances Roles to request COS.
Cvm Role Name	Set the CVM role instance name, see Managing Instances Roles for details.
protocol	Network transfer protocol, which is HTTPS by default. If you want to change it to HTTP, directly modify it in the configuration file.
APPID	APPID is the account you receive after successfully applying for a Tencent Cloud account, automatically assigned by the system, and can be obtained from Account Information . A bucket's full name is composed of two elements: <code>Bucket Name</code> and <code>APPID</code> , formatted as <code><BucketName-APPID></code> , for details please refer to Bucket Naming Conventions .
Bucket Name	Bucket name, together with APPID, constitutes the bucket's full name, formatted as

	<code><BucketName-APPID></code> , for details please refer to Bucket Naming Conventions .
Bucket Endpoint	<p>Domain name of the region where the bucket is located, default domain format is <code>cos.<region>.myqcloud.com</code> , where <code><region></code> represents the bucket's region, such as <code>ap-guangzhou</code>, <code>ap-beijing</code>, etc. For a list of supported regions by COS, see Regions and Access Endpoints.</p> <p>If the bucket has global acceleration enabled, you can configure a global acceleration domain name. For example, the global acceleration domain name is set to <code>cos.accelerate.myqcloud.com</code> ; the internal network global acceleration domain name is set to <code>cos-internal.accelerate.tencentcos.cn</code> .</p>
Bucket Alias	<p>Bucket alias, once configured, you can use <code>BucketAlias</code> to replace <code>BucketName-APPID</code> , reducing the command length required for input. If this option is not configured, the value of <code>BucketAlias</code> is the same as <code>BucketName-APPID</code> .</p>
OFS Bucket	<p>Metadata acceleration bucket tag, used to identify whether the bucket has the metadata acceleration feature enabled.</p>
Auto Switch Host	<p>Set whether to disable automatic switch of a backup domain. Optional values are <code>true</code> and <code>false</code>, which can be null.</p> <p>If the value is not set or set to <code>false</code>, the backup domain switch will be executed;</p> <p>If the value is set to <code>true</code>, the backup domain switch will not be executed.</p>
Disable Encryption	<p>Set whether to disable key encryption. Optional value: <code>true</code> <code>false</code>, which can be left blank.</p> <p>If not set or the value is <code>false</code>, the key-related information in the configuration file will be encrypted.</p> <p>Set to <code>true</code>, and the key-related information in the configuration file will not be encrypted.</p>

During the initial configuration, COSCLI will ask you to configure information for just one bucket. If you want to configure multiple buckets, you can later use the `./coscli config add` command to add bucket configurations. If you need to modify the configuration file or learn more about operations related to the configuration file, refer to the [config command](#) or use the `./coscli config --help` command for quick access to related instructions. Before starting to use commands, you can quickly check how to use COSCLI by running the command `./coscli --help` .

Other configuration methods

In addition to generating the configuration file interactively with the `./coscli config init` command, you can also manually write the COSCLI configuration file. The format of the COSCLI configuration file is in `yaml` format, with an example configuration as follows:

```
cos:
  base:
    secretid: XXXXXXXXXXXXXXXX
```

```
secretkey: XXXXXXXXXXXXXXXXXXXX
sessiontoken: ""
protocol: https
buckets:
- name: examplebucket1-1250000000
  alias: bucket1
  region: ap-shanghai
  endpoint: cos.ap-shanghai.myqcloud.com
  ofs: false
- name: examplebucket2-1250000000
  alias: bucket2
  region: ap-guangzhou
  endpoint: cos.ap-guangzhou.myqcloud.com
  ofs: false
- name: examplebucket3-1250000000
  alias: bucket3
  region: ap-chengdu
  endpoint: cos.ap-chengdu.myqcloud.com
  ofs: false
```

Note:

COSCLI by default reads configuration items from ~/.cos.yaml. If users want to use their own defined configuration file, please use the -c (--config-path) option in the command. The configuration for CFS's secretid/secretkey/sessiontoken are all encrypted strings.

Common Options

Last updated : 2025-05-09 18:27:38

You can view the common options supported by COSCLI with the `./coscli --help` or `./coscli -h` command.

Option Description

The following are common options for COSCLI, which can be used in all its commands:

Note:

We recommend you use a temporary key as instructed in [Generating and Using Temporary Keys](#) to call the SDK for security purposes. When you apply for a temporary key, follow the [Notes on Principle of Least Privilege](#) to avoid leaking resources besides your buckets and objects.

If you must use a permanent key, we recommend you follow the [Notes on Principle of Least Privilege](#) to limit the scope of permission on the permanent key.

Option	Description
-h, --help	Outputs help information. You can view the help information and usage of the tool with the <code>-h</code> or <code>--help</code> command. You can also enter <code>-h</code> after each command (with no parameter appended) to see how to use the command. For example, to view the specific usage of the bucket creation command, enter <code>coscli mb -h</code> .
-c, --config-path	Configuration file path, which is <code>~/.cos.yaml</code> for COSCLI by default. You can also specify a custom configuration file by adding <code>-c</code> after a command.
-e, --endpoint	In addition to configuring the region of a bucket in advance in the configuration file, you can also use <code>-e</code> in COSCLI to specify the bucket endpoint in the format of <code>cos.<region>.myqcloud.com</code> , where <code><region></code> represents the bucket region, such as <code>ap-guangzhou</code> and <code>ap-beijing</code> . For the list of regions supported by COS, see Regions and Access Endpoints .
-i, --secret-id	Specifies the <code>SecretId</code> used to access COS.
-k, --secret-key	Specifies the <code>SecretKey</code> used to access COS.
--session-token	Access COS with a temporary key.
-v, --version	Displays the COSCLI version.
-p, --protocol	Network transfer protocol, which is HTTPS by default.

<code>--init-skip</code>	By default, it is false. If it is set to true (<code>--init-skip=true</code>), skip the config init interactive operation, and directly use the <code>SecretId</code> , <code>SecretKey</code> , and endpoint parameters to request the APIs. When this parameter is used, the <code>-i</code> , <code>-k</code> , and <code>-e</code> parameters must be configured.
<code>--log-path</code>	Custom <code>coscli.log</code> file location, which is in the same directory as COSCLI by default. You can specify a directory or a specific file (the file must end with <code>.log</code>), for example, <code>/data/</code> or <code>/data/coscli.log</code> .

Examples

Example 1: Switching bucket to upload an object

When you need to switch to a bucket in another region through COSCLI, you can use the `-e` option to specify the endpoint of the bucket.

For example, to upload the local file `test.txt` to the bucket `examplebucket-1250000000` in the Chengdu region with the endpoint `cos.ap-chengdu.myqcloud.com`, run the following command:

```
./coscli cp test.txt cos://examplebucket-1250000000/test.txt -e cos.ap-chengdu.myqcloud.com
```

Example 2: Switching user account to view the file list

When you need to use the identity of another account, you can use the `-i` and `-k` options to specify the `SecretId` and `SecretKey` of your key respectively.

For example, to use the identity of another account to list the files in the bucket `examplebucket-1250000000` in the Chengdu region, run the following command:

```
./coscli ls cos://examplebucket-1250000000 -e cos.ap-chengdu.myqcloud.com -i
***** -k *****
```


Common Commands

Generating and Modifying Configuration Files

- config

Last updated : 2025-05-09 18:27:38

Command Syntax

The `config` command is used to generate and modify the configuration file.

```
./coscli config [command] [flag]
```

Note:

After setting the configuration items correctly, you can run `./coscli config show` to view the configuration.

For more general options for this command (such as switching buckets or user accounts), see [Common Options](#).

The generated configuration file uses the HTTPS protocol by default. If you want to change it to HTTP, directly modify it in the configuration file.

`config` includes the following sub-commands:

Command	Description
add	Adds a new bucket configuration.
delete	Deletes an existing bucket configuration.
init	Generates the configuration file interactively.
set	Modifies one or more configuration items in the base group of the configuration file, which contains <code>secretid</code> , <code>secretkey</code> , and <code>sessiontoken</code> .
show	Prints information in a specific configuration file.

`config` and its sub-commands include the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
-c	--config-path	Path of the configuration file to use

The `config add` sub-command includes the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
-a	--alias	Bucket alias
-b	--bucket	Bucket name
-r	--region	Region of the bucket
-o	--ofs	Metadata acceleration bucket flag. For more information, see Metadata Acceleration Overview .

Note:

If you want to specify the endpoint of the bucket, use the common flag `-e` or `--endpoint` . For more information, see [Common Options](#).

The `config delete` sub-command includes the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
-a	--alias	Bucket alias

The `config set` sub-command includes the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
None	--secret_id	Sets the secret ID, which can be created and obtained from the CAM console .
None	--secret_key	Sets the secret key, which can be created and obtained from the CAM console .
-t	--session_token	Sets the temporary key token. For more information on temporary key, see Accessing COS Using a Temporary Key .
None	--mode	Set the identity mode, supporting enumerated values <code>SecretKey</code> and <code>CvmRole</code> . It can be null, with the default value being <code>SecretKey</code> , which means using a key to request

		COS. When the mode is <code>CvmRole</code> , it means requesting COS with managing roles .
None	<code>--cvm_role_name</code>	For the CVM role instance name settings, see managing roles for details.
None	<code>--close_auto_switch_host</code>	Set whether to disable automatic switch of a backup domain. Optional values are true and false, which can be null. If the value is not set or set to false, the backup domain switch will be executed; If the value is set to true, the backup domain switch will not be executed.
None	<code>--disable_encryption</code>	Set whether to disable key encryption. It is false by default. When it is set to true, the key-related information in the configuration file will not be encrypted.

Examples

Adding a new bucket configuration

```
./coscli config add -b examplebucket3-1250000000 -r ap-chengdu -e cos.ap-chengdu.myqcloud.com -a bucket3
```

Deleting an existing bucket configuration

```
./coscli config delete -a bucket3
```

Modifying `session-token` in the default configuration file

```
./coscli config set --session_token test-token123
```

Printing information in a specific configuration file

```
./coscli config show -c /your/config/path.yaml
```

Modifying the mode and the cvmrolename in the default configuration file

```
./coscli config set --mode CvmRole --cvm_role_name testName
```


Creating Buckets - mb

Last updated : 2025-05-09 18:27:38

The `mb` command is used to create a bucket.

Note:

If you need to use this command, when you perform [authorization policy](#), set action to `cos:PutBucket`. For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli mb cos://<BucketName-APPID> -e <endpoint> [flag]
```

Note:

To create a bucket with the `mb` command, you need to add the global flag `-e` or `--endpoint` to specify the region where the bucket resides.

`mb` includes the following parameters:

Parameter Format	Description	Example
cos://<BucketName-APPID>	Customizes the bucket name	cos://examplebucket-1250000000

`mb` includes the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
-r	--region	Indicates the region of the bucket
-m	--maz	Create a multi-az bucket
-o	--ofs	Create a bucket for ofs

Note:

After you run the `mb` command to successfully create a bucket, we recommend that you add information about the bucket in the configuration file, so that you can use the bucket alias for quick operations. For command usage, see the following example.

For more general options for this command (such as switching buckets or user accounts), see [Common Options](#).

Examples

```
// Create the bucket3 bucket.  
./coscli mb cos://bucket3-1250000000 -e cos.ap-chengdu.myqcloud.com
```

If you want to configure an alias for the bucket you just created, update the configuration file with the following command:

```
// Update the configuration file.  
./coscli config add -b bucket3-1250000000 -e cos.ap-chengdu.myqcloud.com -a bucket3  
// After the update, you can access the bucket at cos://bucket3.
```

Deleting Buckets - rb

Last updated : 2025-05-09 18:27:38

The `rb` command is used to delete a bucket.

Note:

If you need to use this command, when you perform [authorization policy](#), set action to `cos:DeleteBucket`. For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli rb cos://<bucket-name> [flag]
```

`rb` includes the following parameters:

Parameter Format	Description	Example
cos://<bucket-name>	Specifies the target bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Access with the bucket alias: cos://example-alias Access with the bucket name: cos://examplebucket-1250000000

`rb` includes the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
-r	--region	Indicates the region of the bucket.

Note:

The `rb` command can only delete empty buckets. If there are files in your bucket, use the `rm` and `abort` commands to clear the files and incomplete multipart uploads in the bucket respectively and then delete the bucket.

After you run the `rb` command to delete the bucket successfully, we recommend that you delete the bucket information in the configuration file. For command usage, see the following example.

For more general options for this command (such as switching buckets or user accounts), see [Common Options](#).

Example

```
// Delete the bucket3 bucket.(require confirmation operation)
./coscli rb cos://bucket3-1250000000 -e cos.ap-chengdu.myqcloud.com

// Update configuration file, delete bucket configuration of bucket3.
./coscli config delete -a bucket3
```


Tagging Bucket - bucket-tagging

Last updated : 2025-05-09 18:27:39

The `bucket-tagging` command is used to create (modify), get, and delete bucket tags. One bucket can have up to 50 tags.

Note:

If you need to obtain bucket tags, when you perform [authorization policy](#), set action to `cos:GetBucketTagging`.

If you need to set the bucket tags, when you perform [authorization policy](#), set action to `cos:PutBucketTagging`.

If you need to delete bucket tags, when you perform [authorization policy](#), set action to

`cos:DeleteBucketTagging`.

For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli bucket-tagging --method [method] cos://<bucket-name> [tag_key]#[tag_value]
```

`bucket-tagging` includes the following parameters:

Parameter Format	Description	Example
<code>cos://<bucket-name></code>	Specifies the target bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Access with the bucket alias: <code>cos://example-alias</code> Access with the bucket name: <code>cos://examplebucket-1250000000</code>

The `bucket-tagging` command contains the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
None	--method	Specifies the operation to be performed, including <code>put</code> (adding tags), <code>get</code> (querying tags), and <code>delete</code> (deleting tags).

Note:

For other common options of this command (such as switching bucket and user account), see [Common Options](#).

Adding or Modifying Bucket Tag

A bucket tag is represented by a key-value pair. Only the bucket owner and users with the `PutBucketTagging` permission can add or modify bucket tags. Error message “403 AccessDenied” will be returned for other users.

Command syntax

```
./coscli bucket-tagging --method put cos://bucketAlias key1#value1 key2#value2
```

Here, `key#value` represents the tag key-value pair separated by `#`. If the bucket does not have a tag, this command will add the specified tag to the bucket; otherwise, it will overwrite the original tag.

Example

To configure two tags with the keys of 1 and 2 as well as values of 111 and 222 respectively for the bucket with alias `example-alias`, run the following command:

```
./coscli bucket-tagging --method put cos://example-alias 1#111 2#222
```

Querying Bucket Tags

Command syntax

```
./coscli bucket-tagging --method get cos://bucketAlias
```

Example

```
./coscli bucket-tagging --method get cos://example-alias
```

The following output shows that the bucket with alias `example-alias` has two tags with the keys of 1 and 2 as well as values of 111 and 222 respectively.

KEY	VALUE
1	111
2	222

Deleting Bucket Tags

Command syntax

```
./coscli bucket-tagging --method delete cos://bucketAlias
```

Example

```
./coscli bucket-tagging --method delete cos://exmaple-alias
```

Querying Bucket/Object List - ls

Last updated : 2025-05-09 18:27:39

The `ls` command is used to query the list of buckets, objects in a bucket, and objects in a directory.

Note:

If you need to list files in the bucket, when you perform [authorization policy](#), set action to `cos:HeadBucket` , `cos:GetBucket` .

If you need to list historical version information (import `--all-versions`), when you perform [authorization policy](#), set action to `cos:HeadBucket` , `cos:GetBucketVersioning` , `cos:GetBucketObjectVersions` .

If you need to list buckets under the account, when you perform [authorization policy](#), set action to `cos:GetService` .

For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli ls [cos://<bucket-name>[/prefix/]] [flag]
```

`ls` includes the following parameters:

Parameter Format	Description	Example
<code>cos://<bucket-name></code>	Specifies the optional target bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Access with the bucket alias: <code>cos://example-alias</code> Access with the bucket name: <code>cos://examplebucket-1250000000</code>
<code>/prefix/</code>	Specifies a directory (optional).	<code>/picture/</code>

`ls` includes the following optional flags:

Flag Abbreviation	Flag Name	Description
<code>-h</code>	<code>--help</code>	Views the usage of this command.
None	<code>--include</code>	Includes specific objects.
None	<code>--exclude</code>	Excludes specific objects.

-r	--recursive	Specifies whether to traverse directories recursively and list all objects.
None	--limit	Specifies the maximum quantity to be listed. (If 0 or no value is configured, the default value is 10000)
None	--all-versions	List all versions of objects. It is available only after version control is enabled for a bucket. Add display fields <code>VersionId</code> , <code>IsLatest</code> and <code>Delete Marker</code> for parameters when listing historical versions.

Note:

`--include` and `--exclude` support standard regular expressions. You can use regular expressions to filter objects that meet your requirements.

When using `zsh` , you may need to add double quotes at both ends of the `pattern` string.

```
./coscli ls cos://bucket1 -r --include ".*\\.mp4$"
```

For other common options of this command (such as switching buckets and user accounts), see [common options](#).

Examples

Listing all buckets of the current account

```
./coscli ls
```

The returned information includes the bucket name, region, creation time, and total number of buckets. Below is an example:

BUCKET NAME	REGION	CREATE DATE
examplebucket-1250000000	ap-nanjing	2022-01-01T00:00:00Z
TOTAL BUCKETS:		2

Listing objects

Listing all objects in the `bucket1` bucket

```
./coscli ls cos://bucket1
```

The returned information includes the object key (the unique identifier of the object in the bucket), storage class, last update time, object size, and total number of objects. Below is an example:

KEY	TYPE	LAST MODIFIED	ETAG
test.txt	STANDARD	2024-06-05T15:03:37+08:00	"a3bc6c9058109f8da48d41a5ab9a"

Listing all objects and subdirectories in the `picture` directory in the `bucket1` bucket

```
./coscli ls cos://bucket1/picture/
```

General listing only returns the data at the level of the query path, without expanding subpaths. Below is an example:

KEY	TYPE	LAST MODIFIED	ET
picture/a4431470f55662	STANDARD	2024-06-05T15:03:58+08:00	"ed0430c5f27e7660"
picture/e98c6cefa4abd6	STANDARD	2024-06-05T15:03:58+08:00	"bd5a4bd7248e7dfd"

Listing all objects in the `picture` directory in the `bucket1` bucket recursively

```
./coscli ls cos://bucket1/picture/ -r
```

If there are subpaths at the level of the query path, recursive listing will scan all subpaths and return all files under the level of the query path. Below is an example:

KEY	TYPE	LAST MODIFIED	
picture/subfolder	DIR		
picture/subfolder/pic2.png	STANDARD	2024-06-05T15:03:58+08:00	"bd5a4bd7248"

Listing all MP4 objects in the `bucket1` bucket recursively

```
./coscli ls cos://bucket1 -r --include ".*\\.mp4$"
```

Listing all non-MP4 objects in the `bucket1` bucket recursively

```
./coscli ls cos://bucket1 -r --exclude ".*\\.mp4$"
```

Listing all non-JPG objects prefixed with `test` in the `picture` directory in the `bucket1` bucket

```
./coscli ls cos://bucket1/picture -r --include "^picture/test.*" --exclude ".*\\.jp"
```

List Historical Versions

List All Earlier Versions in bucket1

```
./coscli ls cos://bucket1/ -r --all-versions
```

If there are subpaths at the level where the query path is located, recursive listing will scan all subpaths and return all historical versions of all files under the level of the query path. An example is as follows:

KEY	TYPE	VERSIONID	ISLATEST
cmd/cmd/abort.go	STANDARD	MTg0NDUwMDM1MjIxMDM3MDI1MDM	false
cmd/cmd/abort_test.go	STANDARD	MTg0NDUwMDM1MjIxMDM3MjgxODI	false
cmd/cmd/bucket_tagging.go	STANDARD	MTg0NDUwMDM1MjIxMDM3MjY3MDI	false
cmd/cmd/bucket_versioning.go	STANDARD	MTg0NDUwMDM1MjIxMDM2MTkzMzU	false
cmd/cmd/buket_tagging_test.go	STANDARD	MTg0NDUwMDM1MjIxMDM2MTY4MDc	false
cmd/cmd/abort.go		MTg0NDUwMDM1MjIwNjc5NTcxMDA	true
cmd/cmd/abort_test.go		MTg0NDUwMDM1MjIwNjcxNzE5NDY	true
cmd/cmd/bucket_tagging.go		MTg0NDUwMDM1MjIwNjc5NjczNzI	true
cmd/cmd/bucket_versioning.go		MTg0NDUwMDM1MjIwNjY3ODc4NzM	true
cmd/cmd/buket_tagging_test.go		MTg0NDUwMDM1MjIwNjY3ODYyMDI	true

Obtaining Statistics on Different Types of Objects - du

Last updated : 2025-05-09 18:27:39

The `du` command is used to list the statistical information of files per storage type under a bucket or a folder. The statistical information includes the total files of different storage types and the total size of the file per type.

Note:

If you need to use this command to stat information of all objects, when you perform [authorization policy](#), set action to `cos:HeadBucket` , `cos:GetBucket` .

If you need to use this command to stat all historical version information, when you perform [authorization policy](#), set action to `cos:HeadBucket` , `cos:GetBucketVersioning` , `cos:GetBucketObjectVersions` .

For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli du cos://<bucket-name>[/prefix/] [flag]
```

`du` includes the following parameters:

Parameter Format	Description	Example
cos://<bucket-name>	Specifies the target bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Access with the bucket alias: cos://example-alias Access with the bucket name: cos://examplebucket-1250000000
/prefix/	Specifies a directory (optional).	/picture/

`du` includes the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
None	--include	Includes specific objects.
None	--exclude	Excludes specific objects.

None	--all-versions	All version data of the statistical object is only available after version control is enabled for the bucket. The statistical information will include the count of DeleteMarkers.
------	----------------	--

Note:

`--include` and `--exclude` support standard regular expression syntax, so you can use them to filter out objects that meet specific criteria.

When using `zsh`, you may need to enclose the pattern string with double quotation marks.

```
./coscli du cos://bucket1/picture/ --include ".*\\.mp4$"
```

For more general options for this command (such as switching buckets or user accounts), see [Common Options](#).

Examples

Listing statistics on objects in the `bucket1` bucket

```
./coscli du cos://bucket1
```

The returned information includes the number and size of objects in different storage classes, total number of objects, and total object size in the bucket. Below is an example:

```

      STORAGE CLASS      | OBJECTS COUNT | TOTAL SIZE
-----+-----+-----
          STANDARD      |             2 |      164 B
        STANDARD_IA     |             0 |           0 B
    INTELLIGENT_TIERING  |             0 |           0 B
          ARCHIVE        |             0 |           0 B
        DEEP_ARCHIVE    |             0 |           0 B
        MAZ_STANDARD     |             0 |           0 B
        MAZ_STANDARD_IA |             0 |           0 B
    MAZ_INTELLIGENT_TIERING |             0 |           0 B
          MAZ_ARCHIVE    |             0 |           0 B
-----+-----+-----
INFO[2022-12-14 17:35:41] Total Objects Count: 2
INFO[2022-12-14 17:35:41] Total Objects Size: 164 B
```

Listing statistics on objects in the `picture` directory in the `bucket1` bucket

```
./coscli du cos://bucket1/picture/
```

Listing statistics on all MP4 objects in the `picture` directory in the `bucket1` bucket

```
./coscli du cos://bucket1/picture/ --include ".*\\.mp4$"
```

Listing statistics on all non-MD objects in the `picture` directory in the `bucket1` bucket

```
./coscli du cos://bucket1/picture/ --exclude ".*\\.md$"
```

List the Files in the `bucket1` bucket and the Statistical Information of All Earlier Versions

```
./coscli du cos://bucket1 --all-versions
```

The returned result is as follows. The output information includes: the number and size of objects under each storage type in the bucket, the total number of objects in the bucket, and the total capacity of objects in the bucket.

STORAGE CLASS	OBJECTS COUNT	TOTAL SIZE
STANDARD	545	1.14 MB
STANDARD_IA	0	0 B
INTELLIGENT_TIERING	0	0 B
ARCHIVE	0	0 B
DEEP_ARCHIVE	13	11.74 KB
MAZ_STANDARD	0	0 B
MAZ_STANDARD_IA	0	0 B
MAZ_INTELLIGENT_TIERING	0	0 B
MAZ_ARCHIVE	0	0 B

```

INFO[2025-02-25 17:36:36] Total Objects Count: 558
INFO[2025-02-25 17:36:36] Total Objects Size: 1.15 MB
INFO[2025-02-25 17:36:36] Total DeleteMarker Count: 501

```

Uploading/Downloading/Copying Objects - cp

Last updated : 2025-06-04 16:13:05

The `cp` command is used to upload, download, or copy objects.

Note:

If you need to use the upload file command, when you perform [authorization policy](#), set action to

`cos:HeadBucket` , `cos:GetBucket` , `cos:HeadObject` , `cos:InitiateMultipartUpload` , `cos:UploadPart` , `cos:CompleteMultipartUpload` , `cos:ListMultipartUploads` , `cos:ListParts` .

If you need to use the download file command, when you perform [authorization policy](#), set action to

`cos:HeadBucket` , `cos:GetBucket` , `cos:HeadObject` , `cos:GetObject` .

If you need to use the copy file command, when you perform [authorization policy](#), set the target object action to

`cos:GetBucket` , `cos:HeadObject` , `cos:InitiateMultipartUpload` , `cos:PutObject` , `cos:CompleteMultipartUpload` . Set the source object action to

`cos:HeadBucket` , `cos:GetBucket` , `cos:HeadObject` , `cos:GetObject` .

For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli cp <source_path> <destination_path> [flags]
```

`cp` includes the following parameters:

Parameter Format	Description	Example
source_path	Source file path, which can be a local path or a COS file path. The COS path is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Local path: ~/example.txt COS file path specified with the bucket alias: cos://bucketalias/example.txt COS file path specified with the bucket name: cos://examplebucket-1250000000/example.txt
destination_path	Destination file path, which can be a local path or a COS file path. The COS path is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Local path: ~/example.txt COS file path specified with the bucket alias: cos://bucketalias/example.txt COS file path specified with the bucket name: cos://examplebucket-1250000000/example.txt

`cp` includes the following optional flags:

Flag Abbreviation	Flag Name	Description
None	<code>--include</code>	Includes specific objects (Versions prior to v1.0.4 only filter the local file name during upload, while versions v1.0.4 and later will filter the full path.)
None	<code>--exclude</code>	Excludes specific objects (Versions prior to v1.0.4 only filter the local file name during upload, while versions v1.0.4 and later will filter the full path.)
<code>-r</code>	<code>--recursive</code>	Specifies whether to traverse all objects in the directory recursively.
None	<code>--storage-class</code>	Specifies the storage class of the uploaded file. Default value: <code>STANDARD</code> . For more information, see Storage Class Overview .
None	<code>--part-size</code>	The part size of the file (default 32 MB, supports up to 5 GB). If you need to adaptively adjust the part size based on file size, set it to 0.
None	<code>--thread-num</code>	Number of concurrent threads. Default value: 5.
None	<code>--rate-limiting</code>	Speed limit for a single URL in MB/s. Value range: 0.1–100 MB/s.
None	<code>--meta</code>	Metadata of the uploaded file, including certain HTTP standard attributes (HTTP Header) and custom metadata prefixed with <code>x-cos-meta-</code> (User Meta). The file metadata is in the format of <code>header:value#header:value</code> , such as <code>Expires:2022-10-12T00:00:00.000Z#Cache-Control:no-cache#Content-Encoding:gzip#x-cos-meta-x:x</code> .
None	<code>--routines</code>	Specifies the number of files for concurrent upload or download of threads between files, with the default number being <code>3</code> .
None	<code>--fail-output</code>	This option determines whether to enable error output of files when uploads or downloads fail (when the default value is <code>true</code> , it is enabled). If it is enabled, failed file transfers will be recorded in the specified directory (if no directory is specified, the default one is <code>./coscli_output</code>). If it is disabled, only the number of failed files will be output to the console.
None	<code>--fail-output-path</code>	This option is used to specify the error output folder for recording failed uploads or downloads. By providing a custom folder path, you can control the location and name of the error output folder. If this option is not set, the default error log folder <code>./coscli_output</code> will be used.
None	<code>--retry-num</code>	Number of frequency limit retry times (default value is <code>0</code> ; No retry). <code>1-10</code>

		times can be selected. When multiple machines execute download operations at the same COS directory simultaneously, you can specify this parameter to retry and avoid frequency limit errors.
None	--err-retry-num	Number of error retry times (default value is <code>0</code>). <code>1-10</code> times are specified, or if the value is set to <code>0</code> , no retry is performed.
None	--err-retry-interval	Retry interval (only available when <code>--err-retry-num</code> is specified as <code>1-10</code>). Specify a retry interval of <code>1-10</code> seconds. If it is not specified or set to <code>0</code> , the value of each retry interval will be random among <code>1-10</code> seconds.
None	--only-current-dir	Whether to only upload files in the current directory; ignore subdirectories and their content (<code>false</code> is set by default; not ignored).
None	--disable-all-symlink	Whether to ignore the subfiles and subdirectories of all the soft links during upload (<code>true</code> is set by default; not uploaded). Currently only supported on Linux and MacOS systems.
None	--enable-symlink-dir	Whether to upload subdirectories of soft links (<code>false</code> is set by default; not uploaded). Currently only supported on Linux and MacOS systems.
None	--disable-crc64	Whether to disable the CRC64 data validation (<code>false</code> is set by default; validation enabled).
None	--disable-checksum	The default value is false, verifying the CRC64 of the entire file. If set to true, only verify shard CRC64.
None	--move	The source file will be deleted after the file is successfully copied to the target path (only available between cos paths).
None	--version-id	Download a specified version file, which is only supported in a Bucket with version control enabled (single file only).

Note:

`cp` automatically uses concurrent upload/download for large objects.

If an object is larger than `--part-size`, COSCLI will split the object into multiple parts according to `--part-size` and use `--thread-num` threads to concurrently upload/download the object.

Each thread maintains a URL. For each URL, you can use the `--rate-limiting` parameter to limit the speed of a single URL. When concurrent upload/download is enabled, the total rate is `--thread-num * --rate-limiting`.

If an object is uploaded/downloaded in parts, checkpoint restart will be enabled by default.

`--include` and `--exclude` support standard regular expression syntax, so you can use them to filter out objects that meet specific criteria.

When using `zsh`, you may need to add double quotes at both ends of the `pattern` string.

```
./coscli cp ~/test/ cos://bucket1/example/ -r --include ".*\\.txt$" --meta=x-cos-m
```

When using commands in Windows CMD, note that the "—" character (Chinese dash) pasted into CMD will automatically change to "--", and you need to manually enter it.

For other common options of this command (such as switching bucket and user account), see [Common Options](#).

Examples

Upload

Uploading a single object

```
./coscli cp ~/example.txt cos://bucket1/example.txt
```

Uploading all files and all folders in the local `test` Directory to the `example` Directory in the `bucket1` bucket

```
./coscli cp ~/test/ cos://bucket1/example/ -r
```

Uploading all .mp4 files under the local `test` folder and its subfolders to the `example` folder in the `bucket1` bucket

```
./coscli cp ~/test/ cos://bucket1/example/ -r --include ".*\\.mp4$"
```

Uploading all non-.md files in the local `test` folder and its subfolders to the `example` folder in `bucket1`

```
./coscli cp ~/test/ cos://bucket1/example/ -r --exclude ".*\\.md$"
```

Uploading all non-.md and non-.html files in the local `test` folder and its subfolders to the `example` folder in the `bucket1` bucket

```
./coscli cp ~/test/ cos://bucket1/example/ -r --exclude ".*\\.html$|.*\\.md$"
```

Uploading all objects in the `dir` directory (containing the `dirA`, `dirB`, `dirC`, and `dirD` subdirectories) except the `dirD` directory

```
./coscli cp dir/ cos://bucket1/example/ -r --exclude "dirD.*"
```

Uploading all files and folders under the local `test` directory to the `example` directory in the `bucket1` bucket, and store them as archive type files

```
./coscli cp ~/test/ cos://bucket1/example/ -r --storage-class ARCHIVE
```

Uploading the local `file.txt` file to the `bucket1` bucket and setting the single-URL speed limit to 1.3 MB/s

```
./coscli cp ~/file.txt cos://bucket1/file.txt --rate-limiting 1.3
```

Download

Downloading a single object

```
./coscli cp cos://bucket1/example.txt ~/example.txt
```

Downloading all files and folders under the `example` directory in the `bucket1` bucket to the local `test` directory

```
./coscli cp cos://bucket1/example/ ~/test/ -r
```

Downloading all .mp4 files in the `example` folder and its subfolders in `bucket1` to the `test` folder on the local device

```
./coscli cp cos://bucket1/example/ ~/test/ -r --include ".*\\.mp4$"
```

Downloading all non-.md files under the `example` directory and its subdirectories in `bucket1` to the local `test` directory

```
./coscli cp cos://bucket1/example/ ~/test/ -r --exclude ".*\\.md$"
```

Downloading all non-.md and non-.html files under the `example` folder and its subfolders in the `bucket1` bucket to the local `test` folder

```
./coscli cp cos://bucket1/example/ ~/test/ -r --exclude ".*\\.html$|.*\\.md$"
```

Download xxx Version of example.txt File in bucket1 to test Directory under Local Device

```
./coscli cp cos://bucket1/example.txt ~/test/ --version-id xxx
```

Copy

Copying a single object within a bucket

```
./coscli cp cos://bucket1/example.txt cos://bucket1/example_copy.txt
```

Copying a single object across buckets

```
./coscli cp cos://bucket1/example.txt cos://bucket2/example_copy.txt
```

Copying all files and folders under the `example1` folder in `bucket1` to the `example2` folder in `bucket2`

```
./coscli cp cos://bucket1/example1/ cos://bucket2/example2/ -r
```

Copying all .mp4 data type files under the `example1` folder and its subfolders in the `bucket1` bucket to the `example2` folder in the `bucket2` bucket

```
./coscli cp cos://bucket1/example1/ cos://bucket2/example2/ -r --include ".*\\.mp4$"
```

Copying all non-.md data type files under the `example1` folder and its subfolders in the `bucket1` bucket to the `example2` folder in the `bucket2` bucket

```
./coscli cp cos://bucket1/example1/ cos://bucket2/example2/ -r --exclude ".*\\.md$"
```

Copy xxx Version of example.txt File in bucket1 to bucket2

```
./coscli cp cos://bucket1/example.txt cos://bucket2/ --version-id xxx
```

Move test Directory in bucket1 to bucket2

```
./coscli cp cos://bucket1/test/ cos://bucket2/test/ --move -r
```


Syncing Upload/Download/Copy - sync

Last updated : 2025-06-04 16:13:05

The `sync` command is used to sync object upload, download, and copy. The difference between `sync` and `cp` is that `sync` first compares the CRC64 value of an object with the same name that already exists, and if the value is the same, the object will not be transferred.

Note:

If you need to use the upload file command, when you perform [authorization policy](#), set action to

`cos:HeadBucket` , `cos:GetBucket` , `cos:HeadObject` , `cos:InitiateMultipartUpload` , `cos:UploadPart` , `cos:CompleteMultipartUpload` , `cos:ListMultipartUploads` , `cos:ListParts` .

If you need to use the download file command, when you perform [authorization policy](#), set action to

`cos:HeadBucket` , `cos:GetBucket` , `cos:HeadObject` , `cos:GetObject` .

If you need to use the copy file command, when you perform [authorization policy](#), set the target object action to

`cos:GetBucket` , `cos:HeadObject` , `cos:InitiateMultipartUpload` , `cos:PutObject` , `cos:CompleteMultipartUpload` . Set the source object action to

`cos:HeadBucket` , `cos:GetBucket` , `cos:HeadObject` , `cos:GetObject` .

For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli sync <source_path> <destination_path> [flag]
```

`sync` includes the following parameters:

Parameter Format	Description	Example
source_path	Source file path, which can be a local path or a COS file path. The COS path is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Local path: ~/example.txt COS file path specified with the bucket alias: cos://bucketalias/example.txt COS file path specified with the bucket name: cos://examplebucket-1250000000/example.txt
destination_path	Destination file path, which can be a local path or a COS file path. The COS path is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in	Local path: ~/example.txt COS file path specified with the bucket alias: cos://bucketalias/example.txt

[Download and Installation Configuration](#). If you use the bucket name for access, you also need to include the `endpoint` flag.

COS file path specified with the bucket name: `cos://examplebucket-1250000000/example.txt`

`sync` includes the following optional flags:

Flag Abbreviation	Flag Name	Description
None	<code>--include</code>	Includes specific objects (Versions prior to v1.0.4 only filter the local file name during upload, while versions v1.0.4 and later will filter the full path.)
None	<code>--exclude</code>	Excludes specific objects (Versions prior to v1.0.4 only filter the local file name during upload, while versions v1.0.4 and later will filter the full path.)
<code>-r</code>	<code>--recursive</code>	Specifies whether to traverse all objects in the directory recursively.
None	<code>--storage-class</code>	Specifies the storage class of the uploaded file. Default value: <code>STANDARD</code> . For more information, see Storage Class Overview .
None	<code>--part-size</code>	The part size of the file (default 32 MB, supports up to 5 GB). If you need to adaptively adjust the part size based on file size, set it to 0.
None	<code>--thread-num</code>	Number of concurrent threads. Default value: 5
None	<code>--rate-limiting</code>	Speed limit for a single URL. Value range: 0.1–100 MB/s
None	<code>--snapshot-path</code>	Specifies the directory where the snapshot information is stored when the uploaded or downloaded file is saved. During the next file upload or download, COSCLI will read the snapshot information in the specified directory for incremental upload or download. This option is used to speed up directory file sync.
None	<code>--meta</code>	Metadata of the uploaded file, including certain HTTP standard attributes (HTTP Header) and custom metadata prefixed with <code>x-cos-meta-</code> (User Meta). The file metadata is in the format of <code>header:value#header:value</code> , such as <code>Expires:2022-10-12T00:00:00.000Z#Cache-Control:no-cache#Content-Encoding:gzip#x-cos-meta-x:x</code> .
None	<code>--routines</code>	Specifies the number of files for concurrent upload or download threads between files. The default value is <code>3</code> .
None	<code>--fail-output</code>	This option determines whether to enable the error output of files when upload or download fails (The default value is <code>true</code> , enabled). If enabled, failed file transfers will be recorded in the specified directory (If no directory is

		specified, the default directory <code>./coscli_output</code> is used). If disabled, only the number of failed files will be output to the console.
None	<code>--fail-output-path</code>	This option is used to specify the error output folder for recording failed file uploads or downloads. By providing a custom folder path, you can control the location and name of the error output folder. If this option is not set, the default error log folder <code>./coscli_output</code> is used.
None	<code>--retry-num</code>	Number of frequency limit retry times (The default value is <code>0</code> , indicating no retry will be performed). It can be specified as <code>1-10</code> times. When multiple machines execute download operations on the same COS directory simultaneously, you can specify this parameter to retry to avoid frequency limit errors.
None	<code>--err-retry-num</code>	Number of error retry times (The default value is <code>0</code>). Set it as <code>1-10</code> times, or set it to <code>0</code> , indicating no retry will be performed.
None	<code>--err-retry-interval</code>	Retry interval (only available when <code>--err-retry-num</code> is set as <code>1-10</code>). Set the retry interval as <code>1-10</code> seconds. If it is not set or is set to <code>0</code> , each retry interval will be a random value between <code>1-10</code> seconds.
None	<code>--only-current-dir</code>	Whether to upload only files in the current directory, ignoring subdirectories and their contents (The default value is <code>false</code> , indicating they will not be ignored).
None	<code>--disable-all-symlink</code>	Whether to ignore the subfiles and subdirectories of all the soft links during upload (The default value is <code>true</code> , indicating they will not be ignored). Currently only supported on Linux and MacOS systems.
None	<code>--enable-symlink-dir</code>	Whether to upload the subdirectories of the soft links (The default value is <code>false</code> , indicating they will not be uploaded). Currently only supported on Linux and MacOS systems.
None	<code>--disable-crc64</code>	Whether to disable CRC64 data validation (The default value is <code>false</code> , indicating validation is enabled).
None	<code>--delete</code>	Deletes any other files in the specified target path, retaining only the files synchronized this time (The default value is <code>false</code> , indicating the files will not be deleted). It is recommended to enable version control before using the <code>--delete</code> option to prevent accidental data deletion.
None	<code>--backup-dir</code>	Synchronizes the backup of deleted files, which preserves files that are deleted on the target end but do not exist on the source end (effective only during the download, and must be specified when <code>--delete=true</code>). For upload and bucket copies, use version control to restore accidentally deleted data.

None	--force	Forced operation, without confirmation prompt (The default value is <code>false</code>).
None	--disable-checksum	The default value is false, verifying the CRC64 of the entire file. If set to true, only verify shard CRC64.

Note:

`sync` automatically uses concurrent upload/download for large objects.

If an object is larger than `--part-size` , COSCLI will split the object into multiple parts according to `--part-size` and use `--thread-num` threads to concurrently upload/download the object.

Each thread maintains a URL. For each URL, you can use the `--rate-limiting` parameter to limit the speed of a single URL. When concurrent upload/download is enabled, the total rate is `--thread-num * --rate-limiting` .

If an object is uploaded/downloaded in parts, checkpoint restart will be enabled by default.

`--include` and `--exclude` support standard regular expression syntax, so you can use them to filter out objects that meet specific criteria.

When using `zsh` , you may need to add double quotes at both ends of the `pattern` string.

Do not set `snapshot-path` to the directory to be migrated or its subdirectories.

```
./coscli sync ~/test/ cos://bucket1/example/ -r --include ".*\\.txt$" --snapshot-
```

When you use the `sync` command, in addition to PUT request fees, there are two other scenarios where you will have a HEAD request for cloud files, which will incur additional fees:

If the parameter to specify the snapshot directory (`--snapshot-path`) is not added, HEAD request fees will be incurred.

If the parameter to specify the snapshot directory (`--snapshot-path`) is added and it is the first time the snapshot directory is generated, HEAD request fees will be incurred. If it is not the first time the snapshot directory is generated, no additional request fees will be incurred.

For other common options of this command (such as switching bucket and user account), see [Common Options](#).

Examples

Syncing object upload

```
./coscli sync ~/example.txt cos://bucket1/example.txt
```

Syncing object download

```
./coscli sync cos://bucket1/example.txt ~/example.txt
```

Syncing intra-bucket replication

```
./coscli sync cos://bucket1/example.txt cos://bucket1/example_copy.txt
```

Syncing cross-bucket replication

```
./coscli sync cos://bucket1/example.txt cos://bucket2/example_copy.txt
```

Deleting Objects - rm

Last updated : 2025-05-09 18:27:39

The `rm` command is used to delete an object.

Note :

To use the `rm` command, please download COSCLI V1.0.1 or above. For details, see [Download and Installation Configuration](#).

If your current version of COSCLI is V1.0.0, please upgrade to V1.0.1 before executing the `rm` command. In V1.0.0, the `--include` and `--exclude` parameters do not work when executing the `rm` command, which may lead to unexpected deletion situations.

If you need to use this command to delete an object, when you perform [authorization policy](#), set action to

```
cos:HeadBucket , cos:HeadObject , cos:GetBucket , cos>DeleteObject ,  
cos>DeleteMultipleObjects .
```

If you need to use this command to delete a historical version (such as importing `--all-versions` or `--version-id`), when you perform [authorization policy](#), set action to `cos:HeadBucket` , `cos:HeadObject` , `cos:GetBucket` , `cos>DeleteObject` , `cos>DeleteMultipleObjects` , `cos:GetBucketVersioning` , `cos:GetBucketObjectVersions` .

For more authorizations, please refer to [CAM-Enabled API](#).If you have any questions or need further assistance, [please contact us](#).

Command Syntax

```
./coscli rm cos://<bucket-name>[/prefix/] [flag]
```

`rm` includes the following parameters:

Parameter Format	Description	Example
cos://<bucket-name>	Specifies the target bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Access with the bucket alias: <code>cos://example-alias</code> Access with the bucket name: <code>cos://examplebucket-1250000000</code>
/prefix/	Specifies a directory (optional).	/picture/

`rm` includes the following optional flags:

--	--	--

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
None	--include	Includes specific objects.
None	--exclude	Excludes specific objects.
-r	--recursive	Specifies whether to traverse all objects in the directory recursively.
-f	--force	Forces deletion (no prompt before the deletion).
None	--fail-output	This option determines whether to enable the error output of files when upload or download fails (The default value is <code>true</code> , enabled). If enabled, failed file transfers will be recorded in the specified directory (If no directory is specified, the default directory <code>./coscli_output</code> is used). If disabled, only the number of failed files will be output to the console.
None	--fail-output-path	This option is used to specify the error output folder for recording failed file uploads or downloads. By providing a custom folder path, you can control the location and name of the error output folder. If this option is not set, the default error log folder <code>./coscli_output</code> is used.
None	--all-versions	Available only in buckets with versioning enabled and when importing the --recursive (-r) parameter, traverse and delete all versions under the specified path.
None	--version-id	Available only in a Bucket with versioning enabled and when --recursive (-r) parameter is not passed in, to delete the specified version of a specified object.

Note:

`--include` and `--exclude` support standard regular expression syntax, so you can use them to filter out objects that meet specific criteria.

When using `zsh` , you may need to add double quotes at both ends of the `pattern` string.

```
./coscli rm cos://bucket1/example/ -r --include ".*\\.mp4$"
```

For more general options for this command (such as switching buckets or user accounts), see [Common Options](#).

Example

Deleting the fig1.png object

```
./coscli rm cos://bucket1/fig1.png
```

Deleting all objects in the `picture` directory

```
./coscli rm cos://bucket1/picture/ -r
```

Deleting the Specified Version of the `fig1.png` File

```
./coscli rm cos://bucket1/fig1.png --version-id xxx
```

Deleting All Versions with the `test` Prefix

```
./coscli rm cos://bucket1/test -r --all-versions
```


Getting File Hash Value - hash

Last updated : 2025-05-09 18:27:39

The `hash` command is used to calculate the hash value of a local file or get the hash value of a file in COS.

Note:

If you need to use this command, when you perform [authorization policy](#), set action to `cos:HeadObject`. For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli hash <object-name> [flag]
```

`hash` includes the following parameters:

Parameter Format	Description	Example
<object-name>	Specifies the target file, which can be a local path or a COS file path. The COS path is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Local path: ~/example.txt COS file path specified with the bucket alias: cos://bucketalias/example.txt COS file path specified with the bucket name: cos://examplebucket-1250000000/example.txt

The `hash` command contains the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
None	--type	Hash type, which can be MD5 or CRC64 (default) Note: The md5 can only obtain the etag. If you need to get the file md5, you can download the complete file and then calculate it.

Note:

For other common options of this command (such as switching bucket and user account), see [Common Options](#).

Examples

Calculating the CRC64 value of local file

```
./coscli hash ~/test.txt
```

Getting the MD5 value of COS file

```
./coscli hash cos://bucket1/example.txt --type=md5
```

Listing Incomplete Multipart Uploads - Lsparts

Last updated : 2025-05-09 18:27:39

The `lsparts` command is used to list the generated incomplete multipart uploads.

Note:

If you need to use this command to list multipart upload tasks, when you perform [authorization policy](#), set action to `cos:PutBucket`.

If you need to use this command to list the fragments of the corresponding file of the multipart upload task, when you perform [authorization policy](#), set action to `cos:ListMultipartUploads`, `cos:ListParts`.

For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli lsparts cos://<bucket-name>[/prefix/] [flag]
```

`lsparts` includes the following parameters:

Parameter Format	Description	Example
<code>cos://<bucket-name></code>	Specifies the target bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Access with the bucket alias: <code>cos://example-alias</code> Access with the bucket name: <code>cos://examplebucket-1250000000</code>
<code>/prefix/</code>	Specifies a directory (optional).	<code>/picture/</code>

The `lsparts` command contains the following optional flags:

Flag Abbreviation	Flag Name	Description
<code>-h</code>	<code>--help</code>	Views the usage of this command.
None	<code>--include</code>	Includes specific objects.
None	<code>--exclude</code>	Excludes files with a specific pattern
None	<code>--limit</code>	Specifies the maximum quantity (0–1000) to be listed.
None	<code>--upload-id</code>	Upload record ID. When uploading the uploadid, you need to

		specify the specific file. Only show the fragment information of the file corresponding to the uploadid. If the uploadid is not passed in, show the parts being uploaded under the specified prefix.
--	--	--

Note:

For more information, see [Multipart Upload](#).

For other common options of this command (such as switching bucket and user account), see [Common Options](#).

Examples

List All Parts Being Uploaded in bucket1

```
./coscli lsparts cos://bucket1
```

The returned information includes the object key (the unique identifier of the object in the bucket), multipart upload ID, multipart upload start time, and total number of incomplete multipart uploads in the bucket. Below is an example:

KEY	UPLOAD ID	INITIATE TIME
test.txt	1671191183635d2b71b1d68a0*****	2022-01-01T00:00:00.000Z
		TOTAL: 1

Listing All Uploaded Fragments of the test.txt File in bucket1 with upload_id 1671191183635d2b71b1d68a0*****

```
./coscli lsparts cos://bucket1/test.txt --upload-id="1671191183635d2b71b1d68a0*****"
```

The returned result is as follows. The output information includes: PARTNUMBER (chunk number), ETAG (MD-5 algorithm checksum of the chunk), LastModified (last modified time of the chunk), SIZE (size of each part).

PARTNUMBER	ETAG	LAST MODIFIED	S
1	"58f06dd588d8ffb3beb46ada6309436b"	2024-12-17T16:34:48+08:00	32.
2	"58f06dd588d8ffb3beb46ada6309436b"	2024-12-17T16:34:48+08:00	32.
3	"58f06dd588d8ffb3beb46ada6309436b"	2024-12-17T16:34:48+08:00	32.
4	"58f06dd588d8ffb3beb46ada6309436b"	2024-12-17T16:34:48+08:00	32.
			TOT

Clearing Incomplete Multipart Uploads - abort

Last updated : 2025-05-09 18:27:39

The `abort` command is used to clear the generated incomplete multipart uploads.

Note:

If you need to use this command, when you perform [authorization policy](#), set action to

`cos:ListMultipartUploads` , `cos:AbortMultipartUpload` . For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli abort cos://<bucket-name>[/prefix/] [flag]
```

`abort` includes the following parameters:

Parameter Format	Description	Example
<code>cos://<bucket-name>/<key></code>	Specifies the target object in the bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Access with the bucket alias: <code>cos://example-alias</code> Access with the bucket name: <code>cos://examplebucket-1250000000</code>
<code>/prefix/</code>	Specifies a directory (optional).	<code>/picture/</code>

The `abort` command contains the following optional flags:

Flag Abbreviation	Flag Name	Description
<code>-h</code>	<code>--help</code>	Views the usage of this command.
None	<code>--include</code>	Includes specific objects
None	<code>--exclude</code>	Excludes files with a specific pattern
None	<code>--fail-output</code>	This option determines whether to enable error output when cleaning up file fragments (enabled by default <code>true</code>). If enabled, file fragment cleanup failures will be

		recorded in the specified directory (if not specified, defaults to <code>./coscli_output</code>). If disabled, only the number of cleanup failures will be output to the console.
None	--fail-output-path	This option is used to specify the error output folder for recording during file fragment cleanup. By providing a custom folder path, you can control the location and name of the error output folder. If this option is not set, the default error log folder <code>./coscli_output</code> will be used.

Note:
For other common options of this command (such as switching bucket and user account), see [Common Options](#).

Examples

Clearing all incomplete multipart uploads in `bucket1` bucket

```
./coscli abort cos://bucket1
```

Clearing all incomplete multipart uploads in `picture` folder in `bucket1` bucket

```
./coscli abort cos://bucket1/picture/
```

Retrieving Archived Files - restore

Last updated : 2025-05-09 18:27:39

The `restore` command is used to restore archived objects.

Note:

If you need to use this command, when you perform [authorization policy](#), set action to `cos:HeadBucket` , `cos:GetBucket` , `cos:PostObjectRestore` . For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli restore cos://<bucket-name>[/prefix/] [flag]
```

`restore` includes the following parameters:

Parameter Format	Description	Example
cos://<bucket-name>	Specifies the target bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Access with the bucket alias: cos://example-alias Access with the bucket name: cos://examplebucket-1250000000
/prefix/	Specifies a directory (optional).	/picture/

The `restore` command contains the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
None	--include	Includes specific objects.
None	--exclude	Excludes specific objects.
-d	--days	Specifies the expiration time of the temporary file generated during object restoration, which is 3 days by default.
-m	--mode	Specifies the restoration mode, which is <code>Standard</code> by default

		<p>If the data to be recovered is of the archive storage type, the available values are Expedited, Standard, and Bulk, which correspond to expedited retrieval mode, standard retrieval mode, and bulk retrieval mode respectively.</p> <p>If the data being restored is of deep archive storage type, the optional values are Standard, Bulk.</p>
-r	--recursive	Traverses the folder recursively.
None	--fail-output	This option determines whether to enable the error output of files when reheating fails (enabled by default <code>true</code>). If enabled, the failure of file reheating will be recorded in the specified directory (if not specified, defaults to <code>./coscli_output</code>). If disabled, only the quantity of error files will be output to the console.
None	--fail-output-path	This option is used to specify the error output folder for recording failed reheating files. By providing a custom folder path, you can control the location and name of the error output folder. If this option is not set, the default error log folder <code>./coscli_output</code> will be used.

Note:

`--include` and `--exclude` support standard regular expression syntax, so you can use them to filter out objects that meet specific criteria.

When using `zsh`, you may need to add double quotes at both ends of the `pattern` string.

```
./coscli restore cos://bucket1/example/ -r --include ".*\\.mp4$"
```

For more general options for this command (such as switching buckets or user accounts), see [Common Options](#).

For more information on retrieving archived files, see [POST Object restore](#).

Examples

Restoring archived objects in `bucket1` bucket in Standard mode

```
./coscli restore cos://bucket1/picture.jpg
```

Restoring all archived objects in `picture` folder in `bucket1` bucket in Expedited mode

```
./coscli restore cos://bucket1/picture/ -r --mode Expedited
```

Creating/Obtaining a Symbolic Link - symlink

Last updated : 2025-05-09 18:27:39

The symlink command is used to create or obtain a symbolic link to an object. You can quickly find the object through this symbolic link.

Note:

If you need to use the command to create a symbolic link, when you perform [authorization policy](#), set action to `cos:PutBucket` .

If you need to use the command to obtain a symbolic link, when you perform [authorization policy](#), set action to `cos:GetObject` .

For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli symlink --method create cos://<bucket-name>/<key> --link linkKey [flag]
```

The symlink command includes the following parameters:

Parameter Format	Description	Example
cos://<bucket-name>	Specifies the target bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Access with the bucket alias: cos://example-alias Access with the bucket name: cos://examplebucket-1250000000
/<key>	Specifies the object name.	/test

The symlink command includes the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
No	--method	Valid values: create/get.
No	--link	Specifies the symbolic link key value.

Operation Example

Creating a symbolic link named symlink for picture.jpg in the bucket1 bucket

```
./coscli symlink --method create cos://bucket1/picture.jpg --link symlink
```

Accessing the target object via the symlink

```
./coscli symlink --method get cos://bucket2 --link symlink
```

Viewing Contents of an Object - cat

Last updated : 2025-05-09 18:27:40

The cat command is used to view the contents of an object.

Note:

If you need to use this command, when you perform [authorization policy](#), set action to `cos:GetBucket`. For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli cat cos://<bucket-name>/<key> [flag]
```

The cat command includes the following parameters:

Parameter Format	Description	Example
cos://<bucket-name>/<key>	Specifies the target bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the endpoint flag.	Access with the bucket alias: cos://example-alias/test.txt Access with the bucket name: cos://examplebucket-1250000000/test.txt

The cat command includes the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.

Operation Example

Viewing the contents of the test.txt file in the bucket1 bucket

```
./coscli cat cos://bucket1/test.txt
```

Getting Pre-signed URL - signurl

Last updated : 2024-01-06 16:15:35

The `signurl` command is used to get the pre-signed URL of an object, through which the object can be accessed anonymously.

Command Syntax

```
./coscli signurl cos://<bucket-name>/<key> [flag]
```

`signurl` includes the following parameters:

Parameter Format	Description	Example
cos://<bucket-name>/<key>	Specifies the target object in the bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Access with the bucket alias: cos://example-alias/test.txt Access with the bucket name: cos://examplebucket-1250000000/test.txt

The `signurl` command contains the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
-t	--time	Sets the URL expiration time, which is 1000s by default

Note:

For other common options of this command (such as switching bucket and user account), see [Common Options](#).

Examples

Getting the pre-signed URL of `picture.jpg` in `bucket1` bucket

```
./coscli signurl cos://bucket1/picture.jpg
```

Getting the pre-signed URL of `picture.jpg` in `bucket2` bucket and setting the URL expiration time to 1314s

```
./coscli signurl cos://bucket2/picture.jpg --time 1314
```

Listing Contents and Statistics Under a Directory - lsdu

Last updated : 2025-05-09 18:27:40

The lsdu command is used to obtain the contents of the current level with a specified prefix and to calculate the size and number of objects within it.

Note:

If you need to use this command, when you perform [authorization policy](#), set action to

`cos:HeadBucket` , `cos:GetBucket` . For more authorizations, please refer to [CAM-Enabled API](#).

Command Syntax

```
./coscli lsdu cos://<bucket-name>[/prefix/] [flags]
```

The lsdu command includes the following parameters:

Parameter Format	Description	Example
cos://<bucket-name>	Specifies the target bucket, which is accessible by using the bucket alias or bucket name configured in the configuration file as detailed in Download and Installation Configuration . If you use the bucket name for access, you also need to include the <code>endpoint</code> flag.	Access with the bucket alias: cos://example-alias Access with the bucket name: cos://examplebucket-1250000000
/prefix/	Specifies a directory (optional).	/picture/

The lsdu command includes the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
No	--include	Includes specific objects.
No	--exclude	Excludes specific objects.

Operation Example

Obtaining the statistics under the root directory of the bucket1 bucket

```
./coscli lsdu cos://bucket1/
```

The returned information includes the names of directories or objects, the total number of objects, and the total size.

Below is an example:

NAME	OBJECTS COUNT	TOTAL SIZE
-----+-----+-----		
300123/	1	300.00 MB
300s/	1	100.00 MB
300u/	301	8.22 GB
activity/	35	129.08 KB
test/	1	3 B
test100/	20	20.00 GB
test5/	6	9.00 GB
testrm/	1	0 B
10GB_file	1	11.72 GB
1GB_file	1	1.00 GB

Obtaining the statistics under the picture directory of the bucket1 bucket

```
./coscli lsdu cos://bucket1/picture/
```

Note:

When there are many files, this command may take a long time to run. It is recommended to run it in the background.

Bucket Versioning - bucket-versioning

Last updated : 2025-05-09 18:27:40

The bucket-versioning command is used to manage versioning for a bucket. It can be used to enable, suspend, and view the status of bucket versioning.

Note:

If you need to use the command to enable/suspend versioning, when you perform [authorization policy](#), set action to `cos:PutBucketVersioning`.

If you need to use the command to view the bucket versioning status, when you perform [authorization policy](#), set action to `cos:GetBucketVersioning`.

For more authorizations, please see [CAM-Enabled API](#).

Command Syntax

The following bucket-versioning commands are used to enable and suspend versioning:

```
./coscli bucket-versioning --method [method] cos://<bucket-name> versioning
```

The bucket-versioning and its subcommands include the following optional flags:

Flag Abbreviation	Flag Name	Description
-h	--help	Views the usage of this command.
-c	--method	Valid values: put/get.

Operation Example

Enable Bucket Versioning

```
./coscli bucket-versioning --method put cos://examplebucket Enabled
```

Suspend Bucket Versioning

```
./coscli bucket-versioning --method put cos://examplebucket Suspended
```

View Bucket Versioning Status

```
./coscli bucket-versioning --method get cos://examplebucket
```

FAQs

Last updated : 2024-01-06 16:15:35

What's the difference between COSCLI and COSCMD?

1. COSCLI is a binary package compiled using Golang. It is ready-to-use and does not require installing any dependency before the installation and deployment. However, COSCMD is compiled in Python and requires installing the Python environment and dependency packages.
2. COSCLI supports setting an alias for a bucket, meaning that you can use a short string to represent `<BucketName-APPID>` for convenience. However, COSCMD does not support setting aliases and requires using `<BucketName-APPID>` to specify a bucket, making the commands harder to use and read.
3. COSCLI supports configuring multiple buckets in the configuration file and supports cross-bucket operations. However, with COSCMD, you can only configure one bucket in the configuration file and the commands for cross-bucket operations are long.

COSCMD

Last updated : 2025-05-30 17:35:03

Feature Overview

COSCMD enables you to use simple command lines to batch-operate objects, such as upload, download, and delete.

Operating Environments

Operating system

Windows, Linux, and macOS.

Note:

Local characters should use UTF-8 encoding. Otherwise, exceptions will occur when you operate on Chinese files. Ensure that the local time is in sync with UTC. If there is a large deviation between the two, COSCMD might not function properly.

Software dependency

Python 2.7 and Python 3

Latest version of pip

Note:

It is recommended that users install the Python version with integrated pip directly.

Installation and configuration

For more information about the installation and configuration of the environment, see [Python](#).

For more information about the installation and configuration of pip, see [Installation](#).

Download and Installation

You can install COSCMD in the following three ways:

Method 1:

1. Installing with pip

Run the `pip` command to install COSCMD:

```
pip install coscmd
```

After the installation is complete, you can run the `-v` or `--version` command to view the version information.

Note:

After installing COSCMD in Windows, you need to add the `C:\\python_install_dir` and `C:\\python_install_dir\\Scripts` paths to environment variables.

2. Upgrading with pip

After the installation is complete, you can run the following command to upgrade COSCMD:

```
pip install coscmd -U
```

Method 2: Installing with the source code (not recommended)

You can click [here](#) to download the source code.

```
git clone https://github.com/tencentyun/coscmd.git
cd coscmd
python setup.py install
```

Note:

If your Python version is 2.6, installing the dependent libraries with pip may fail. Therefore, this installation method is recommended.

Method 3: Installing from offline

Note:

Ensure that the two devices are installed with the same version of Python. Otherwise, the installation might fail.

```
# Run the following commands on a device that is connected to the Internet
mkdir coscmd-packages
pip download coscmd -d coscmd-packages
tar -czvf coscmd-packages.tar.gz coscmd-packages

# Copy the installation package to a device that is not connected to the Internet
tar -xzvf coscmd-packages.tar.gz
pip install coscmd --no-index -f coscmd-packages
```

Parameter Configuration

Viewing the help option

Run the `-h` or `--help` command to view the information and usage of COSCMD.

```
coscmd -h
```

The help information is as follows:

```
usage: coscmd [-h] [-d] [-s] [-b BUCKET] [-r REGION] [-c CONFIG_PATH]
             [-l LOG_PATH] [--log_size LOG_SIZE]
             [--log_backup_count LOG_BACKUP_COUNT] [-v]
             {config,upload,download,delete,abort,copy,move,list,listparts,info,restore,signurl,createbucket,deletebucket,putobjectacl,getobjectacl,putbucketacl,getbucketacl,putbucketversioning,getbucketversioning,probe}
```

an easy-to-use but powerful command-line tool. try 'coscmd -h' to get more informations. try 'coscmd sub-command -h' to learn all command usage, likes 'coscmd upload -h'

positional arguments:

{config,upload,download,delete,abort,copy,move,list,listparts,info,restore,signurl,createbucket,deletebucket,putobjectacl,getobjectacl,putbucketacl,getbucketacl,putbucketversioning,getbucketversioning,probe}	
config	Config your information at first
upload	Upload file or directory to COS
download	Download file from COS to local
delete	Delete file or files on COS
abort	Aborts upload parts on COS
copy	Copy file from COS to COS
move	move file from COS to COS
list	List files on COS
listparts	List upload parts
info	Get the information of file on COS
restore	Restore
signurl	Get download url
createbucket	Create bucket
deletebucket	Delete bucket
putobjectacl	Set object acl
getobjectacl	Get object acl
putbucketacl	Set bucket acl
getbucketacl	Get bucket acl
putbucketversioning	Set the versioning state
getbucketversioning	Get the versioning state
probe	Connection test

optional arguments:

-h, --help	show this help message and exit
-d, --debug	Debug mode
-s, --silence	Silence mode
-b BUCKET, --bucket BUCKET	Specify bucket
-r REGION, --region REGION	

```

Specify region
-c CONFIG_PATH, --config_path CONFIG_PATH
Specify config_path
-l LOG_PATH, --log_path LOG_PATH
Specify log_path
--log_size LOG_SIZE    specify max log size in MB (default 1MB)
--log_backup_count LOG_BACKUP_COUNT
                        specify log backup num
-v, --version          show program's version number and exit

```

In addition, you can enter `-h` after each command (with no parameter appended) to see how to use the command.

For example:

```
coscmd upload -h // View how to use the upload command
```

Generating a configuration file

COSCMD will first read the necessary information from the configuration file before running. By default, COSCMD will read configuration items from `~/.cos.conf`.

Note:

Before the configuration, you need to go to the COS console to create a bucket (e.g., `configure-bucket-1250000000`) for parameter configuration and create a key.

The following is a sample configuration file:

```

[common]
secret_id = AKIDA6wUmImTMzvXZNbGLCgtusZ2E8mG****
secret_key = TghWBCyf5LIyTcXCoBdw1oRpytWk****
bucket = configure-bucket-1250000000
region = ap-chengdu
max_thread = 5
part_size = 1
retry = 5
timeout = 60
schema = https
verify = md5
anonymous = False

```

Note:

The `timeout` item in the configuration file, in seconds, indicates the timeout period for reading and writing data.

`schema` can be set to `http` or `https` (default).

`anonymous` can be set to `True` or `False`. If it is set to `True`, the anonymous mode is used (the signature is empty).

For more information about the parameter description, run the `coscmd config -h` command.

Generating a configuration file via the `config` command

Note:

We recommend you use a temporary key as instructed in [Generating and Using Temporary Keys](#) to call the SDK for security purposes. When you apply for a temporary key, follow the [Notes on Principle of Least Privilege](#) to avoid leaking resources besides your buckets and objects.

If you must use a permanent key, we recommend you follow the [Notes on Principle of Least Privilege](#) to limit the scope of permission on the permanent key.

You can run the `config` command to automatically generate a configuration file in `~/.cos.conf`. The command format is as follows:

```
coscmd config [OPTION]...<FILE>...
    [-h] --help
    [-a] <SECRET_ID>
    [-s] <SECRET_KEY>
    [-t] <TOKEN>
    [-b] <BucketName-APPID>
    [-r] <REGION> | [-e] <ENDPOINT>
    [-m] <MAX_THREAD>
    [-p] <PART_SIZE>
    [--do-not-use-ssl]
    [--anonymous]
```

Note:

Fields enclosed in "[]" are optional, and those in "<>" are required.

The parameters are described as follows:

Option	Parameter Description	Valid Value	Required
-a	Key ID, which can be obtained at Manage API Key	String	Yes
-s	Key, which can be obtained at Manage API Key	String	Yes
-t	Temporary key token, which needs to be specified in the <code>x-cos-security-token</code> header when a temporary key is used.	String	No
-b	Name of the specified bucket, formatted as <code>BucketName-APPID</code> . For more information, see Naming Conventions . If this is your first time using COSCMD, you need to create a bucket in the COS console to configure COSCMD.	String	Yes
-r	Region of the bucket. For more information, see Regions and Access Endpoints .	String	Yes
-e	Endpoint of the request. Once you configure this parameter, the	String	No

	region parameter will be invalidated. If you use the default endpoint, this parameter is formatted as <code>cos.<region>.myqcloud.com</code> ; if you use a global acceleration endpoint, the format is <code>cos.accelerate.myqcloud.com</code> .		
-m	Maximum number of threads for multithreaded operation (default is 5). If the file upload is slow, you can increase this value. The number of threads depends on the machine's performance. Assuming the machine supports a maximum of 30 threads, you can adjust the concurrent threads to 30 to fully utilize the machine performance. The command is <code>coscmd config -m 30</code> .	Number	No
-p	Size of the multipart upload part, in MB (default: 1; value range: 1-1000)	Number	No
--do-not-use-ssl	Uses the HTTP protocol instead of HTTPS	String	No
--anonymous	Anonymous operation (without carrying a signature)	String	No

The following sample shows how to use the `config` command:

```
coscmd config -a AChT4ThiXAbpBDEFGhT4ThiXAbp**** -s WE54wreefvds3462refgwewe**** -b
```

Common Commands

Specifying a bucket and its region

If you do not specify the bucket and region in the commands, the commands take effect for the bucket that is used to configure COSCMD. To perform operations on another bucket, you need to specify the bucket and the region where the bucket resides.

Note:

Use the `-b <BucketName-APPID>` parameter to specify the bucket name, which must be formatted as `BucketName-APPID` .

Use the `-r <region>` parameter to specify the region where the bucket resides.

Command syntax

```
coscmd -b <BucketName-APPID> -r <region> <action> ...
```

Sample: creating a bucket named `examplebucket` that resides in the Beijing region

```
coscmd -b examplebucket-1250000000 -r ap-beijing createbucket
```

Sample: uploading “picture.jpg” from D drive to the `examplebucket` bucket

```
coscmd -b examplebucket-1250000000 -r ap-beijing upload D:/picture.jpg /
```

Specifying the configuration file and log file paths

If you do not specify the configuration file path, `~/.cos.conf` will be used by default. Similarly, if you do not specify the log file path, `~/.cos.log` will be used by default.

Note:

Use the `-c <conf_path>` parameter to specify the configuration file path. COSCMD will read configuration information from the path when running.

Use the `-l <log_conf>` parameter to specify the log file path. COSCMD will output the logs generated at runtime to the log files in the path.

Command syntax

```
coscmd -c <conf_path> -l <log_conf> <action> ...
```

Sample: setting the configuration file's path to `/data/home/cos_conf` and the log path to `/data/home/cos_log`, and creating a bucket named `examplebucket` in the Beijing region.

```
coscmd -c /data/home/cos_conf -l /data/home/cos_log -b examplebucket-1250000000 -r
```

Running commands in debugging mode

If `-d` or `-debug` is added before a command, detailed operation information will be displayed when executing the command, as shown in the example below:

Command syntax

```
coscmd -d upload <localpath> <cospath>
```

Sample: outputting detailed information upon the upload

```
coscmd -d upload -rs D:/folder/ /
```

Running commands in silence mode

You can prefix `-s` or `--silence` in each command so that no message will be output.

Note:

To run this command, the version should be at least 1.8.6.24.

Command syntax

```
coscmd -s upload <localpath> <cospath>
```

Sample:

```
coscmd -s upload D:/picture.jpg /
```

Common Bucket Commands

Creating a bucket

Note:

Specify `-b <BucketName-APPID>` and `-r <Region>` when you run the `coscmd createbucket` command; otherwise, an error may be reported. This is because if the bucket and region are not specified, this command takes effect for the bucket that is used to configure COSCMD.

Command syntax

```
coscmd -b <BucketName-APPID> createbucket
```

Sample: creating a bucket named `examplebucket` that resides in the Beijing region

```
coscmd -b examplebucket-1250000000 -r ap-beijing createbucket
```

Deleting a bucket

Note:

`coscmd deletebucket` takes effect only for the bucket that is used to configure COSCMD. To delete another bucket, run the command with the `-b <BucketName-APPID>` and `-r <region>` parameters specified.

Command syntax

```
coscmd -b <BucketName-APPID> deletebucket
```

Sample: deleting empty buckets

```
coscmd -b examplebucket-1250000000 -r ap-beijing deletebucket
```

Sample: forcibly deleting a non-empty bucket

```
coscmd -b examplebucket-1250000000 -r ap-beijing deletebucket -f
```

Note:

The `-f` parameter will forcibly delete the bucket, including all files, noncurrent folders (if versioning is enabled), and incomplete multipart uploads.

Common Object Commands

Uploading files

Command syntax for uploading a file

```
coscmd upload <localpath> <cospath>
```

Note:

Replace "localpath" and "cospath" enclosed in "<>" with the path of the local file to upload and the COS storage path, respectively.

Sample: uploading "picture.jpg" in D drive to the "doc" folder of COS

```
coscmd upload D:/picture.jpg doc/
```

Sample: uploading "picture.jpg" in the "doc" folder in D drive to the "doc" folder of COS

```
coscmd upload D:/doc/picture.jpg doc/
```

Sample: uploading a file to the ARCHIVE storage class to the "doc" directory of COS

```
coscmd upload D:/picture.jpg doc/ -H '{"x-cos-storage-class':'Archive'}"
```

Note:

When you set the HTTP header with the `-H` parameter, please use the JSON format, for example, `coscmd upload -H '{"x-cos-storage-class':'Archive','Content-Language':'zh-CN'}' <localpath> <cospath>`. For more information about the headers, see [PUT Object](#).

Sample: setting meta attributes and uploading a file to the "doc" folder of COS

```
coscmd upload D:/picture.jpg doc/ -H '{"x-cos-meta-example':'example'}"
```

Sample: Set the upload speed limit to 800Kb/s and upload the file file.zip from the doc folder on Drive D to the doc directory on COS.

```
coscmd upload D:/doc/file.zip doc/ -H '{"x-cos-traffic-limit':'819200'}"
```

Note:

Specify the request header x-cos-traffic-limit via the -H parameter to limit the speed. The speed range is 819200 to 838860800 (in bit/s), that is, 800 Kb/s to 800 Mb/s. If a value is not within this range, a 400 error will be returned.

Uploading a folder

Command syntax for uploading a folder

```
coscmd upload -r <localpath> <cospath>
```

Note:

Windows users are advised to use the `upload` command in cmd or PowerShell that comes with the system. Other tools, such as Git Bash, have a different command path resolution strategy than PowerShell and can cause users' files to be uploaded to an incorrect path.

Sample: uploading the "doc" folder in D drive to the root directory of COS

```
coscmd upload -r D:/doc /
```

Sample: uploading the "doc" folder in D drive to the "doc" folder of COS

```
coscmd upload -r D:/doc doc
```

Sample: uploading files synchronously (files with the same name, MD5 checksum, and file size will be skipped)

```
coscmd upload -rs D:/doc doc
```

Note:

Use the `-s` parameter to upload files synchronously while skipping those with the same MD5 value (please note that the source files in COS must have been uploaded using COSCMD v1.8.3.2 or above; the `x-cos-meta-md5` header is included by default).

Sample: uploading files synchronously (files with the same name and file size will be skipped)

```
coscmd upload -rs --skipmd5 D:/doc doc
```

Note:

The `-s` parameter allows synchronous upload, and the `--skipmd5` parameter can be used to skip files with the same name and same file size.

Sample: uploading the folder synchronously and deleting files that are deleted in the "doc" folder in D drive

```
coscmd upload -rs --delete D:/doc /
```

Note:

For example, if the files in the Drive D doc folder have been synchronously uploaded to the doc path in COS on the same day, and the source and destination are consistent. On the next day, if the user deletes file A from the Drive D doc folder, executing the above command will delete file A from the doc path in COS, ultimately keeping the files in the Drive D doc folder consistent with those in the doc path in COS.

Sample: ignoring .txt and .doc files in the "doc" folder in D drive

```
coscmd upload -rs D:/doc / --ignore *.txt,*.doc
```

Sample: ignoring .txt files in the "doc" folder in D drive

```
coscmd upload -rs D:/doc / --ignore "*.txt"
```

Note:

When uploading folders, you can ignore certain types of files by using the `--ignore` parameter, or filter certain types of files by using `--include`. Multiple shell wildcard rules (separated by commas `,`) are supported. When ignoring a certain suffix, it must be enclosed in `" "`.

If you want to use `--ignore` to filter all files in a particular folder, you need to use an absolute path and use `" "` to enclose the path, for example, `coscmd upload -rs D:/doc / --ignore "D:/doc/ignore_folder/*"`.

Sample: uploading .txt and .doc files in the "doc" folder in D drive

```
coscmd upload -rs D:/doc / --include *.txt,*.doc
```

Sample: uploading .txt files in the "doc" folder in D drive

```
coscmd upload -rs D:/doc / --include "*.txt"
```

Note:

If the file to upload is larger than 10 MB, COSCMD will upload it with multipart upload. The command is `coscmd upload <localpath> <cospath>` (same as simple upload).

COSCMD supports checkpoint restart to resume the upload of large files. When the multipart upload of a large file fails, only the failed parts will be uploaded when the operation is resumed instead of starting over from scratch (please ensure that the directory and content of the re-uploaded file are consistent with the uploaded directory).

COSCMD performs MD5 verification on each part during multipart upload.

When COSCMD uploads a file, the `x-cos-meta-md5` header is carried by default, whose value is the file's MD5 checksum. If the command has carried the `--skipmd5` parameter, this header will not be carried.

Querying a file list

The query command is as follows:

Command syntax

```
coscmd list <cospath>
```

Sample: recursively querying the list of all files prefixed with "doc/" in this bucket

```
coscmd list doc/
```

Sample: recursively querying the file list, number of files, and the file sizes of a bucket.

```
coscmd list -ar
```

Sample: recursively querying the list of all files prefixed with "examplefolder"

```
coscmd list examplefolder/ -ar
```

Sample: querying the historical versions of all files in a bucket

```
coscmd list -v
```

Note:

Replace "cospath" enclosed in "<>" with the COS path of the file list to query. If `<cospath>` is empty, the root directory of the current bucket is queried.

Use `-a` to query all files.

Use `-r` to query files recursively. The number and total size of the files are listed at the end of the returned result.

Use `-n num` to set the maximum number of files to query.

Viewing the file information

The command is as follows:

Command syntax

```
coscmd info <cospath>
```

Sample: viewing the metadata of "doc/picture.jpg"

```
coscmd info doc/picture.jpg
```

Note:

Replace "cospath" enclosed in "<>" with the COS path of the file to display.

Downloading a file or folder

Command syntax for downloading a file

```
coscmd download <cospath> <localpath>
```

Note:

Replace "cospath" and "localpath" enclosed in "<>" with the COS path of the file to download and the local storage path, respectively.

Sample: downloading the "doc/picture.jpg" file in COS to "D:/picture.jpg"

```
coscmd download doc/picture.jpg D:/picture.jpg
```

Sample: downloading the "doc/picture.jpg" file in COS to drive D

```
coscmd download doc/picture.jpg D:/
```

Sample: downloading a specified version of "picture.jpg" to drive D

```
coscmd download picture.jpg --versionId MTg0NDUxMzc2OTM4NTExNTg7Tjg D:/
```

Command syntax for downloading a folder

```
coscmd download -r <cospath> <localpath>
```

Sample: downloading the "doc" folder to "D:/folder/doc"

```
coscmd download -r doc D:/folder/
```

Sample: downloading files in the root directory while ignoring those in the `doc` directory that is under the root directory

```
coscmd download -r / D:/ --ignore "doc/*"
```

Sample: downloading all files in the root directory of the current bucket and overwriting local files

```
coscmd download -rf / D:/examplefolder/
```

Note:

If a file with the same name exists locally, the download will fail. In this case, you need to use the `-f` parameter to overwrite the local file.

Sample: synchronously downloading all files in the root directory of the current bucket while skipping those with the same filename and MD5 checksum

```
coscmd download -rs / D:/examplefolder
```

Note:

Use the `-s` or `--sync` parameter to skip identical files that already exist locally when downloading a folder (provided that the downloaded files were uploaded via the COSCMD `upload` API and the `x-cos-meta-md5` header was included).

Sample: synchronously downloading all files in the root directory of the current bucket while skipping those with the same filename and file size

```
coscmd download -rs --skipmd5 / D:/examplefolder
```

Sample: Synchronous download and delete "files deleted from the doc path in the COS bucket"

```
coscmd download -rs --delete / D:/doc /
```

Note:

For example, if the files in the D drive doc folder and the files under the doc path in COS have been synchronously uploaded on the same day, making the source end and target end identical. The next day, if the user deletes file A under the doc path in COS, executing the above command will delete file A from the D drive doc folder, ultimately keeping the files in the D drive doc folder and the files under the doc path in COS consistent.

Sample: ignoring .txt or .doc files


```
coscmd download -rs / D:/examplefolder --ignore *.txt,*.doc
```

Sample: ignoring .txt files

```
coscmd download -rs / D:/examplefolder --ignore "*.txt"
```

Note:

When uploading folders, you can ignore certain types of files by using the `--ignore` parameter, or filter certain types of files by using `--include`. Multiple shell wildcard rules (separated by commas `,`) are supported. When ignoring a certain suffix, it must be enclosed in `" "`.

If you want to use `--ignore` to filter all files in a particular directory, you need to use an absolute path and use

`" "` to enclose the path, for example, `coscmd upload -rs D:/doc / --ignore "D:/doc/ignore_folder/*"`.

Sample: filtering .txt and .doc files

```
coscmd download -rs / D:/examplefolder --include *.txt,*.doc
```

Sample: filtering .txt files

```
coscmd download -rs / D:/examplefolder --include "*.txt"
```

Note:

Since the old `mget` API is disused, please use the `download` API for multipart downloads.

Getting signed download URLs

Command syntax

```
coscmd signurl <cospath>
```

Sample: generating a signed URL for "doc/picture.jpg"

```
coscmd signurl doc/picture.jpg
```

Sample: generating a signed URL that is effective for 100 seconds for "doc/picture.jpg"

```
coscmd signurl doc/picture.jpg -t 100
```

Note:

Replace "cospath" enclosed in "<>" with the COS path of the file for which you need to get the download URL.

The `-t time` parameter sets the effective time (in seconds) of the URL signature. The default value is 10000.

Deleting a file or folder

Command syntax for deleting a file

```
coscmd delete <cospath>
```

Note:

Replace "cospath" enclosed in "<>" with the COS path of the file to delete. You will be prompted to confirm this operation.

Sample: deleting the "doc/exampleobject.txt" file

```
coscmd delete doc/exampleobject.txt
```

Sample: deleting files with version IDs

```
coscmd delete doc/exampleobject.txt --versionId MTg0NDUxMzc4ODA3NTgyMTErEWN
```

Command syntax for deleting a folder

```
coscmd delete -r <cospath>
```

Sample: deleting the `doc` folder

```
coscmd delete -r doc
```

Sample: deleting the `folder/doc` folder

```
coscmd delete -r folder/doc
```

Sample: deleting all files with version IDs in the `doc` directory

```
coscmd delete -r doc/ --versions
```

Note:

You need to enter `y` to confirm a batch delete operation. You can skip this step if the `-f` parameter is used.

Note that the delete folder command will delete the current folder as well as the files in it. To delete a versioning-enabled file, you need to specify a version ID.

Viewing incomplete multipart uploads

Command syntax

```
coscmd listparts <cospath>
```

Sample: listing incomplete multipart uploads prefixed with "doc/"

```
coscmd listparts doc/
```

Aborting incomplete multipart uploads

Command syntax

```
coscmd abort
```

Sample: aborting all incomplete multipart uploads

```
coscmd abort
```

Copying a file or folder

Command syntax for copying a file

```
coscmd copy <sourcepath> <cospath>
```

Sample (intra-bucket replication): copying "picture.jpg" in the `examplebucket-1250000000` bucket to the "doc" folder

```
coscmd -b examplebucket-1250000000 -r ap-chengdu copy examplebucket-1250000000.ap-c
```

Sample (cross-bucket replication): copying "doc/picture.jpg" in the `examplebucket2-1250000000` bucket to "doc/examplefolder/" in the `examplebucket1-1250000000` bucket.

```
coscmd -b examplebucket1-1250000000 -r ap-guangzhou copy examplebucket2-1250000000.
```

Change the storage class of the file to STANDARD_IA.

```
coscmd -b examplebucket1-1250000000 -r ap-guangzhou copy examplebucket2-1250000000.
```

Change the storage class of the file to ARCHIVE and rename it "photo.jpg".

```
coscmd -b examplebucket1-1250000000 -r ap-guangzhou copy examplebucket2-1250000000.
```

Command syntax for copying a folder

```
coscmd copy -r <sourcepath> <cospath>
```

Sample: copying the `examplefolder` directory in the `examplebucket2-1250000000` bucket to the `doc` directory in the `examplebucket1-1250000000` bucket

```
coscmd -b examplebucket1-1250000000 -r ap-guangzhou copy -r examplebucket2-12500000
```

Note:

Replace "sourcepath" and "cospath" enclosed in "<>" with the path of the COS file to copy and the COS destination path, respectively.

The source path is formatted as `<BucketName-APPID>.cos.<region>.myqcloud.com/<cospath>` .

Use the `-d` parameter to set the `x-cos-metadata-directive` header. Valid values are `Copy` (default) and `Replaced` .

When setting the HTTP header with the `-H` parameter, use the JSON format, for example, `coscmd copy -H -d Replaced '{"x-cos-storage-class':'Archive','Content-Language':'zh-CN'}"`
`<localpath> <cospath>` . For more information about the headers, see [PUT Object - Copy](#).

Moving a file or folder

Note:

In a `move` command, `<sourcepath>` and `<cospath>` cannot be the same. Otherwise, files will be deleted. The reason is that the `move` command copies files first and then deletes them, and the files in the `<sourcepath>` path are eventually deleted.

Command syntax for moving a file

```
coscmd move <sourcepath> <cospath>
```

Sample (intra-bucket movement): moving "picture.jpg" in the `examplebucket-1250000000` bucket to the "doc" folder

```
coscmd -b examplebucket-1250000000 -r ap-chengdu move examplebucket-1250000000.ap-c
```

Sample (cross-bucket movement): moving "picture.jpg" in the `examplebucket2-1250000000` bucket to "doc/folder/" in the `examplebucket1-1250000000` bucket

```
coscmd -b examplebucket1-1250000000 -r ap-guangzhou move examplebucket2-1250000000.
```

Sample: changing the storage class of the file to STANDARD_IA

```
coscmd -b examplebucket1-1250000000 -r ap-guangzhou move examplebucket2-1250000000.
```

Sample: changing the storage class of the file to ARCHIVE

```
coscmd -b examplebucket1-1250000000 -r ap-guangzhou move examplebucket2-1250000000.
```

Command syntax for moving a folder

```
coscmd move -r <sourcepath> <cospath>
```

Sample: moving the "examplefolder" directory in the `examplebucket2-1250000000` bucket to the "doc" directory in the `examplebucket1-1250000000` bucket

```
coscmd -b examplebucket1-1250000000 -r ap-guangzhou move -r examplebucket2-12500000
```

Note:

Replace "sourcepath" and "cospath" enclosed in "<>" with the path of the COS file to move and the COS destination path, respectively.

The source path is formatted as `<BucketName-APPID>.cos.<region>.myqcloud.com/<cospath>` .

Use the `-d` parameter to set the `x-cos-metadata-directive` header. Valid values are `Copy` (default) and `Replaced` .

When setting the HTTP header with the `-H` parameter, use the JSON format, for example, `coscmd move -H -d Replaced '{"x-cos-storage-class': 'Archive', 'Content-Language': 'zh-CN'}"` `<localpath> <cospath>` . For more information about the headers, see [PUT Object - Copy](#).

Setting object access permission

Command syntax

```
coscmd putobjectacl --grant-<permissions> <UIN> <cospath>
```

Sample: granting `100000000001` permission to read "picture.jpg"

```
coscmd putobjectacl --grant-read 100000000001 picture.jpg
```

Sample: querying the file's access permission

```
coscmd getobjectacl picture.jpg
```

Enabling/Suspending versioning

Command syntax

```
coscmd putbucketversioning <status>
```

Sample: enabling versioning

```
coscmd putbucketversioning Enabled
```

Sample: suspending versioning

```
coscmd putbucketversioning Suspended
```

Sample: querying versioning

```
coscmd getbucketversioning
```

Note:

Replace "status" enclosed in "<>" with the desired versioning status.

Once versioning is enabled for the bucket, it cannot return to the prior status (initial status). However, you can suspend versioning for the bucket so that subsequent uploads of objects will not generate multiple versions.

Restoring an ARCHIVED file

Command syntax for restoring an archived file

```
coscmd restore <cospath>
```

Sample: restoring "picture.jpg" using the expedited retrieval mode (effective for 3 days)

```
coscmd restore -d 3 -t Expedited picture.jpg
```

Command syntax for restoring archived files

```
coscmd restore -r <cospath>
```

Sample: restoring the "examplefolder/" folder using the expedited retrieval mode (effective for 3 days)

```
coscmd restore -r -d 3 -t Expedited examplefolder/
```

Note:

Replace "cospath" enclosed in "<>" with the COS path of the file list to query.

Use `-d <day>` to set the validity period of the temporary copy. Default value: `7`.

Use `-t <tier>` to specify the restoration mode. Enumerated values: `Expedited` (Expedited retrieval), `Standard` (Standard retrieval), and `Bulk` (bulk retrieval). Default value: `Standard`.

FAQs

If you have any questions about COSCMD, see [COSCMD](#).

COS Migration

Last updated : 2024-04-30 16:42:14

Feature Overview

COS Migration is an all-in-one tool that integrates the COS data migration feature. You can migrate local data to COS through simple configurations and steps. It has the following features:

Checkpoint restart: Restarting uploads from checkpoints is supported. For large files, if the upload exits halfway or service failure occurs, you can run the tool again to restart the upload.

Multipart upload: An object can be uploaded to COS by parts.

Parallel upload: Multiple objects can be uploaded at the same time.

Note:

COS Migration only supports UTF-8 encoding.

If you use this tool to upload a file that already has the same name, the existing file will be overwritten. You need to configure the tool to skip files with the same name.

Use the migration service platform preferably for scenarios other than local data migration.

COS Migration is used for **one-time** migration but is not suitable for continuous sync. For example, if files are added locally every day and need to be continuously synced to COS, then in order to avoid repeated migration tasks, COS Migration will save the records of successful migrations. In case of continuous sync, the record scanning time will keep increasing. We recommend you use COSBrowser as described in [User Guide for Desktop Version](#) for this scenario.

Operating Environments

Operating system

Windows and Linux.

Software dependency

JDK 1.8 X64 or above. For more information, see [Java Installation and Configuration](#).

IFUNC needs to be supported on Linux and the binutils version should be later than 2.20.

How to Use

1. Get the tool

Download COS Migration [here](#).

2. Decompress the package

Windows

Decompress the package and save it to a directory, for example:

```
C:\\Users\\Administrator\\Downloads\\cos_migrate
```

Linux

Decompress the package and save it to a directory, for example:

```
unzip cos_migrate_tool_v5-master.zip && cd cos_migrate_tool_v5-master
```

Migration tool structure

The structure of the properly decompressed COS Migration tool is as follows:

```
COS_Migrate_tool
|--conf #Directory of the configuration file
|   |--config.ini #Migration configuration file
|--db #Store the record of successful migrations
|--dep #JAR package complied by the main logic of the program
|--log #Log generated during tool execution
|--opbin #Script for compiling
|--result #A directory used to save successful migration records. The record file i
|--src #Source code of the tool
|--tmp #Temporary file storage directory
|--.gitignore #Files and folders ignored by the Git version controller.
|--pom.xml #Project configuration file
|--README #Readme document
|--start_migrate.sh #Migration startup script for Linux
|--start_migrate.bat #Migration startup script for Windows
```

Note:

The `db` directory mainly records the IDs of files successfully migrated by the tool. Each migration job will first compare the records in the `db` directory. If the ID of the current file has already been recorded, the current file will be skipped, otherwise it will be migrated.

The `log` directory keeps all the logs generated during tool migration. If an error occurs during migration, first check `error.log` in this directory.

3. Modify the config.ini file

Before running the migration startup script, modify the config.ini file (path: `./conf/config.ini`) first. This file contains the following parts:

3.1 Configure the migration type

type indicates the migration type, which is fixed to `type=migrateLocal` .

```
[migrateType]
type=migrateLocal
```

3.2 Configure the migration job

You can configure a migration job based on your actual needs, including information configuration for the destination COS and job-related configurations.

```
# The common configuration section of the migration tool includes account informati
[common]
secretId=COS_SECRETID
secretKey=COS_SECRETKEY
bucketName=examplebucket-1250000000
region=ap-guangzhou
storageClass=Standard
cosPath=/
https=off
tmpFolder=./tmp
smallFileThreshold=5242880
smallFileExecutorNum=64
bigFileExecutorNum=8
entireFileMd5Attached=on
executeTimeWindow=00:00,24:00
outputFinishedFileFolder=./result
resume=false
skipSamePath=false
requestTryCount=5
```

Name	Description	Default Value
secretId	SecretId of your key. Replace <code>COS_SECRETID</code> with your real key information, which can be obtained on the TencentCloud API key page in the CAM console	-
secretKey	SecretKey of your key. Replace <code>COS_SECRETKEY</code> with your real key information, which can be obtained on the TencentCloud API key page in the CAM console	-
bucketName		-

	Name of the destination bucket in the format of <code><BucketName-APPID></code> . The bucket name must include the APPID such as <code>examplebucket-1250000000</code>	
region	Region information of the destination bucket. For the region abbreviations in COS, see Regions and Access Domain Names	-
storageClass	Storage class for the migrated data. Valid values: <code>Standard</code> , <code>Standard_IA</code> , <code>Archive</code> . For more information, see Storage Class Overview .	Standard
cosPath	COS path to migrate to. <code>/</code> indicates to migrate to the root path of the bucket, <code>/folder/doc/</code> indicates to migrate to <code>/folder/doc/</code> in the bucket. If <code>/folder/doc/</code> does not exist, a path will be created automatically	/
https	Whether to transfer via HTTPS. on: Yes, off: No. It takes time to enable transfer via HTTPS, which is suitable for scenarios that demand high security.	off
tmpFolder	The directory used to store temporary files when data is migrated from another cloud storage service to COS, which will be deleted after the migration is completed. The format must be an absolute path: The separator on Linux is <code>/</code> , such as <code>/a/b/c</code> The separator on Windows is <code>\\</code> , such as <code>E:\\\\a\\\\b\\\\c</code> The default value is the tmp directory in the path of the tool	./tmp
smallFileThreshold	Number of bytes as the threshold for small files. If the size is greater than or equal to this threshold, multipart upload is used; otherwise, simple upload is used. The default value is 5 MB (5242880 Byte)	5242880
smallFileExecutorNum	Concurrency for uploading small files (smaller than <code>smallFileThreshold</code>) via simple upload. Decrease the concurrency if files are uploaded to COS via public network with low bandwidth	64
bigFileExecutorNum	Concurrency for uploading large files (greater than or equal to <code>smallFileThreshold</code>) via multipart upload. Decrease the concurrency if files are uploaded to COS via public network with low bandwidth	8
entireFileMd5Attached	The migration tool calculates the MD5 of the entire file and stores it in the custom header "x-cos-meta-md5" of the file for subsequent verification, because the ETag of a large file uploaded to COS via multipart upload is not the MD5 of the entire file	on
executeTimeWindow	Execution time window, at a granularity of minutes. This parameter	00:00,24:00

	<p>defines the daily execution time range of the migration tool. For example:</p> <p>The parameter "03:30,21:00" means that tasks are executed between 03:30 and 21:00. Outside these hours, the migration tasks enter a sleep state. In sleep state, migration is paused but the progress is retained, and migration is automatically resumed at the next time window. Note that the end time point must be later than the start time.</p>	
outputFinishedFileFolder	<p>This directory stores the results of successful migration tasks, and result files are named by date, for example, <code>./result/2021-05-27.out</code>, where <code>./result</code> is the directory that is created. Each line in the result files is in the format of <code>"Absolute path"\t"File size"\t"Last modified time"</code>. If <code>outputFinishedFileFolder</code> is left empty, no results will be output.</p>	./result
resume	Whether to continue with the result of the last run and traverse through the list of files from the source. The tool starts from scratch by default.	false
skipSamePath	Whether to skip the current file if a file with the same name already exists in COS. By default, the tool does not skip the current file: it overwrites the existing file.	false
requestTryCount	Total number of attempts for each file upload.	5

3.3 Configure the data source

3.3.1 Configure a local data source migrateLocal

If you migrate from a local system to COS, configure this section. The specific configuration items and descriptions are as follows:

```
# Configuration section for migration from a local system to COS
[migrateLocal]
localPath=E:\\\\code\\\\java\\\\workspace\\\\cos_migrate_tool\\\\test_data
excludes=
ignoreModifiedTimeLessThanSeconds=
```

Configuration Item	Description
localPath	<p>Absolute path of the local directory</p> <p>Linux uses a slash (/) as the delimiter, for example, <code>/a/b/c</code>.</p> <p>Windows uses two backslashes (\\) as the delimiter, for example, <code>E:\\\\a\\\\b\\\\c</code>.</p>

	Note: You can enter only a directory path but not file path for this parameter; otherwise, an error will occur while parsing the target object name. In the case of <code>cosPath=/</code> , the request will be incorrectly parsed into a bucket creation request.
<code>excludes</code>	Absolute path of the directory or file to be excluded, meaning some directories or files under <code>localPath</code> are not to be migrated. Multiple absolute paths are separated by semicolons. If this is left blank, all files in <code>localPath</code> will be migrated
<code>ignoreModifiedTimeLessThanSeconds</code>	Exclude files that have an update time less than a certain period of time from the current time (in seconds). This item is left blank by default, indicating files are not to be filtered by the time specified by <code>lastmodified</code> . It is suitable for scenarios where you run the migration tool while updating files and don't want files being updated to be migrated to COS. For example, if it is configured as <code>300</code> , only files updated at least 5 minutes ago will be uploaded.

4. Run the migration tool

Windows

Double-click **start_migrate.bat** to run the tool

Linux

1. Read the configuration from the `config.ini` file by running the following command:

```
sh start_migrate.sh
```

2. Read the configuration from command lines for some parameters by running the following command:

```
sh start_migrate.sh -Dcommon.cosPath=/savepoint0403_10/
```

Note:

The tool supports reading configuration items in two ways: command line or configuration file.

The command line takes priority over the configuration file, i.e., for the same configuration item, parameters in command lines take priority.

Reading configuration items from command lines allows users to run different migration jobs at the same time, provided that key configuration items (such as bucket name, COS path, source path to be migrated, etc.) in the two jobs are not exactly the same. Concurrent migration can be achieved because different migration jobs are written into different `db` directories. Refer to `db` information in the tool structure above.

Configuration items are in the format of **-D{sectionName}.{sectionKey}={sectionValue}**. `sectionName` is the section name of the configuration file. `sectionKey` is the name of the configuration item in the section.

`sectionValue` is the value of the configuration item in the section. COS path to which data is migrated to should be in the format of **-Dcommon.cosPath=/bbb/ddd**.

Migration mechanism and process

Migration mechanism

The COS migration tool is stateful. Files successfully migrated are recorded in the db directory, stored in the LevelDB files in the form of KV. Before each migration task check whether the path exists in the db directory. If the path is available and "mtime" is consistent with that in the db directory, the migration task is skipped. Otherwise, the migration task proceeds. Therefore, you need to check whether there are successful migration records in the db directory, rather than searching in COS. If the migration tool is bypassed and files are deleted or modified through other means (such as COSCMD or the console), such change will not be detected after the migration tool runs and migration will not be performed again.

Migration process

1. The configuration file is read, the corresponding configuration section is read according to the migration type, and parameters are checked.
2. Scan and compare the identifiers for the files to be migrated in the db directory to check whether the files can be uploaded.
3. During the migration execution process, the execution result is printed, where "inprogress" means migrating, "skip" means skipped, "fail" means failure, "ok" means success, and "condition_not_match" indicates files skipped due to not meeting migration conditions (such as lastmodified and excludes). Details of the failure can be found in the error log of the log.
4. Statistics are printed out after the migration is completed, which include the total number of migrated, failed, and skipped files as well as the amount of time consumed. For failures, check the error log, or rerun the migration job as the migration tool will skip successfully migrated files and retry migrating failed ones. The execution result of a migration job is shown below:

```
migrateAli over! op statistics:
      op_status : ALL_OK
      migrate_ok : 530038
      migrate_fail : 0
      migrate_skip : 496264
      start_time : 2018-03-19 15:52:02
      end_time : 2018-03-19 16:54:38
      used_time : 3756 s
```

FAQs

If an exception such as migration failure or execution error occurs when you use the COS Migration, troubleshoot as instructed in [COS Migration](#).

Conclusion

Besides, COS offers not only the aforementioned applications and services but also a variety of popular open-source applications integrated with Tencent Cloud COS plugins. You are welcome to click [here](#) to launch them instantly with a single click.

FTP Server

Last updated : 2024-01-06 16:15:34

Feature Overview

COS FTP Server allows you to directly operate on COS objects and directories over FTP protocol, including uploading/downloading/deleting files and creating folders. This tool is developed with Python, which makes the installation easier.

Feature Overview

Upload mechanism: streaming without saving uploaded files locally. It works as long as the working directory is configured using the standard FTP protocol, and no actual disk storage capacity is occupied.

Download mechanism: the downloaded file is directly streamed and returned to the client.

Directory mechanism: bucket serves as the root directory of the entire FTP Server, and multiple subdirectories can be created under a bucket.

Binding to multi-buckets: multiple buckets can be bound at the same time.

Note:

Binding to multi-buckets: Multiple buckets can be bound via different FTP Server working paths (`home_dir`). Therefore, ensure a unique `home_dir` is assigned to each bucket and user.

Restriction on deletion: the `delete_enable` option can be configured for each FTP user in the new FTP Server to identify whether the FTP user is allowed to delete files.

Supported FTP commands: `put` , `mput` , `get` , `rename` , `delete` , `mkdir` , `ls` , `cd` , `bye` , `quite` , and `size` .

Unsupported FTP commands: `append` and `mget` (The native `mget` command is not supported, but batch download is allowed on certain Windows clients, such as the FileZilla client.)

Note:

The FTP Server tool does not support checkpoint restart.

Getting Started

Operating system

OS: Linux. The [CVM](#) of Tencent CentOS series is recommended. Windows systems are not supported for now.

psutil-dependent Linux package: python-devel or python-dev, depending on the names of different Linux distributions. It is added using Linux package manager, such as `yum install python-devel` or `aptitude install python-dev`.

Python interpreter version: Python 2.7. For more information on the installation and configuration, see [Python](#).

Note:

FTP Server does not support Python 3.

Dependent packages:

[cos-python-sdk-v5](#) (≥1.6.5)

[pyftplib](#) (≥1.5.2)

[psutil](#) (≥5.6.1)

Use restrictions

Applicable to COS XML version.

Installation and Operation

Download the FTP Server tool at [cos-ftp-server](#). The installation steps are as follows:

1. Enter the FTP Server directory, and run `setup.py` to install FTP Server and its dependent libraries (network required).

```
python setup.py install    # Your account should use sudo or have the root
                             permission.
```

2. Copy the configuration sample file `conf/vsftpd.conf.example` and name it `conf/vsftpd.conf`. See [Configuration File](#) of this document to correctly configure bucket and user information.

3. Run `ftp_server.py` to start FTP Server.

```
python ftp_server.py
```

FTP Server can also be started in the following two ways:

Execute the `nohup` command to start it in the backend process:

```
nohup python ftp_server.py >> /dev/null 2>&1 &
```

Execute the `screen` command to run it at the backend (you need to install the screen tool):

```
screen -dmS ftp
screen -r ftp
python ftp_server.py
#Use the keyboard shortcut Ctrl + A + D to go back to the main screen.
```

Stop Operation

If FTP Server is running directly or running at the backend with the `screen` command, you can stop it with the shortcut key combination `Ctrl+C`.

If FTP Server is started with the `nohup` command, you can stop it by the following way:

```
ps -ef | grep python | grep ftp_server.py | grep -v grep | awk '{print $2}' |
xargs -I{} kill {}
```

Configuration File

The configuration sample file for FTP Server is `conf/vsftpd.conf.example`. Copy and name it `vsftpd.conf`, and then configure it as follows:

```
[COS_ACCOUNT_0]
cos_secretid = COS_SECRETID      # Replaced with your SECRETID
cos_secretkey = COS_SECRETKEY    # Replaced with your SECRETKEY
cos_bucket = examplebucket-1250000000
cos_region = region             # Replaced with your bucket region
cos_protocol = https
#cos_endpoint = region.myqcloud.com
home_dir = /home/user0          # Replaced with the local path you want the FTP to mount
ftp_login_user_name=user0       # Replaced with a custom username
ftp_login_user_password=pass0   # Replaced with a custom password
authority=RW                    # The user's read and write permissions. R: read; W: write
delete_enable=true              # true allows the FTP user to delete files by default; false prohibits

[COS_ACCOUNT_1]
cos_secretid = COS_SECRETID      # Replaced with your SECRETID
cos_secretkey = COS_SECRETKEY    # Replaced with your SECRETKEY
cos_bucket = examplebucket-1250000000
cos_region = region             # Replaced with your bucket region
cos_protocol = https
#cos_endpoint = region.myqcloud.com
home_dir = /home/user1          # Replaced with the local path you want the FTP to mount
ftp_login_user_name=user1       # Replaced with a custom username
ftp_login_user_password=pass1   # Replaced with a custom password
authority=RW                    # The user's read and write permissions. R: read; W: write
delete_enable=false             # true allows the FTP user to delete files by default; false prohibits

[NETWORK]
# If the FTP Server is behind a gateway or NAT, you can use this section to specify
masquerade_address = XXX.XXX.XXX.XXX
# The listening port for FTP Server is 2121 by default. Please note that your WAF must allow
```

```
listen_port = 2121
# `passive_port` sets the available port range in Passive mode, with a default of [
passive_port = 60000,65535

[FILE_OPTION]
# By default, the maximum size of a single file is 200 G. We do not recommend going
single_file_max_size = 21474836480

[OPTIONAL]
# For the following settings, take the default settings unless otherwise needed. Fi
min_part_size      = default
upload_thread_num  = default
max_connection_num = 512
max_list_file      = 10000          # The maximum number of files to be list
log_level          = INFO           # Set the log output level.
log_dir            = log            # Set the directory to store logs. Defau
```

Note:

To bind each user to a unique bucket, simply add the `[COS_ACCOUNT_X]` section.

The section for each `COS_ACCOUNT_X` is described as follows:

The username (`ftp_login_user_name`) and the home directory (`home_dir`) under each account must be unique, and the home directory must be a directory that exists in the system.

The number of users logging in to each COS FTP Server simultaneously cannot exceed 100.

`endpoint` and `region` will not take effect at the same time. To use the public cloud COS service, enter the `region` field correctly. The `endpoint` is commonly used in the privatized deployment environment. When both `region` and `endpoint` are entered, `endpoint` will take precedence.

The OPTIONAL part in the configuration file is used to adjust the upload performance for advanced users. You can obtain an optimal uploading speed by reasonably adjusting the part size and the number of concurrent upload threads based on the server performance. General users can keep the default settings without adjustment.

Meanwhile, the limit option for the maximum number of connections is provided. If you do not want to set a limit to it, enter 0, meaning no limit to the maximum number of connections (a reasonable evaluation is required based on your server performance).

Generally, for the `masquerade_address` section in your configuration file, we recommend you specify the IP address that your client is using to connect to the COS FTP Server. If you have any questions, please see the FAQs about [FTP Server](#).

Assume that the FTP Server has more than one IP address, and after running the `ifconfig` command, you get a private ENI IP `10.xxx.xxx.xxx`, which is mapped to the public IP `119.xxx.xxx.xxx`. At this time, if the FTP Server does not explicitly set `masquerade_address` to the public IP (119.xxx.xxx.xxx) that the client uses to access the server, the FTP Server in Passive mode may use the private IP (10.xxx.xxx.xxx) to return packets to the client. As a result, the client is able to connect to the FTP Server, but cannot return data packets to the client properly.

Therefore, generally speaking, we recommend you to set `masquerade_address` to the IP address that your client is using to connect to the Server.

In the configuration file, `listen_port` sets the listening port for the COS FTP Server, and is defaulted to 2121.

`passive_port` sets the range of data channel listening ports for the COS FTP Server, and is defaulted to [60000, 65535]. When your client connects to the COS FTP Server, ensure that your WAF allows the ports configured in these two sections.

Quick Practice

Accessing COS FTP Server using Linux `ftp` command

1. Install the Linux FTP client.

```
yum install -y ftp
```

2. Open the Linux command line, and use the command `ftp [ip address] [port No.]` to connect to the COS FTP Server. Example:

```
ftp 192.xxx.xx.103 2121
```

In the `ftp` command, the IP field corresponds to the **masquerade_address** section in the sample configuration file `conf/vsftpd.conf.example`. In this example, the IP is set to `192.xxx.xx.103`.

In the `ftp` command, the port field corresponds to the **listen_port** section in the sample configuration file `conf/vsftpd.conf.example`. In this example, the port is set to `2121`.

3. Run the above command, and you can see **Name** and **Password** to be entered. Copy and paste the contents of the `ftp_login_user_name` and `ftp_login_user_password` sections for COS FTP Server, and the connection will succeed.

Name: Corresponds to **ftp_login_user_name** (requires configuration) in the sample configuration file

`conf/vsftpd.conf.example`.

Password: Corresponds to **ftp_login_user_password** (requires configuration) in the sample configuration file

`conf/vsftpd.conf.example`.

Accessing COS FTP Server using FileZilla

1. Download and install [FileZilla client](#).

2. After configuring the access information for COS FTP Server on your FileZilla client, click **Quick Connect**.

Host (H): corresponds to **masquerade_address** in the sample configuration file

`conf/vsftpd.conf.example`. In this example, the IP is set to `192.xxx.xx.103`.

Note:

If the COS FTP Server is behind a gateway or NAT, you can use this section to specify the gateway's IP address or domain name as the Server's IP address.

Username (U): Corresponds to **ftp_login_user_name** (requires configuration) in the sample configuration file

```
conf/vsftpd.conf.example .
```

Password (W): Corresponds to **ftp_login_user_password** (requires configuration) in the sample configuration file

```
conf/vsftpd.conf.example .
```

Port (P): Corresponds to **listen_port** in the sample configuration file `conf/vsftpd.conf.example` . In this example, the port is set to `2121` .

FAQs

If any error occurs or you have any question on the upload limit while using FTP Server, see [FTP Server FAQs](#).

Hadoop

Last updated : 2024-01-06 16:15:35

Feature Overview

Hadoop-COS implements a standard Hadoop file system on the Tencent Cloud COS platform. It helps integrate COS with big data computing frameworks such as Hadoop, Spark, and Tez, so that they can read and write COS data as they do with HDFS file systems.

Since Hadoop-COS uses COSN (a Tencent Cloud big data tool) as its URI scheme, it can also be referred to as a COSN-based file system.

Operating Environments

Operating system

Windows, Linux, or macOS

Software dependency

Hadoop-2.6.0 or later

Note:

Hadoop-COS has been integrated in Apache Hadoop-3.3.0. For details, click [here](#).

If your version is earlier than Apache Hadoop-3.3.0, or the CDH has integrated the Hadoop-COS JAR package, you need to restart NodeManager to load the JAR package.

To build a JAR package of a specified Hadoop version, modify `hadoop.version` in the pom file.

Download and Installation

Obtaining the Hadoop-COS JAR package and dependencies

Download link: [Hadoop-COS release](#)

Installing the Hadoop-COS plugin

1. Copy `hadoop-cos-{hadoop.version}-{version}.jar` and `cos_api-bundle-{version}.jar` to `$HADOOP_HOME/share/hadoop/tools/lib`.

Note:

Select the JAR package that corresponds to your Hadoop version. If you cannot find the desired JAR package in the release, manually build and generate one by modifying `hadoop.version` in the pom file.

2. Modify the `hadoop-env.sh` file under the `$HADOOP_HOME/etc/hadoop` directory by adding the COSN JAR package to your Hadoop environment variables as follows:

```
for f in $HADOOP_HOME/share/hadoop/tools/lib/*.jar; do
  if [ "$HADOOP_CLASSPATH" ]; then
    export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$f
  else
    export HADOOP_CLASSPATH=$f
  fi
done
```

Configuration Method

COSN configuration

Attribute Key	Description	Default Value
fs.cosn.userinfo.secretId/secretKey	The API key for your account. Log in to the CAM console to view the key.	None
fs.cosn.credentials.provider	<p>The way to get SecretId and SecretKey. Currently, the following ways are supported:</p> <ol style="list-style-type: none"> 1. org.apache.hadoop.fs.auth.SessionCredentialProvider: Gets them from the request URI in the format of cosn://{secretId}:{secretKey}@examplebucket-1250000000/. 2. org.apache.hadoop.fs.auth.SimpleCredentialProvider: Gets them by reading fs.cosn.userinfo.secretId and fs.cosn.userinfo.secretKey in the core-site.xml configuration file. 3. org.apache.hadoop.fs.auth.EnvironmentVariableCredentialProvider: Gets them from system variables COS_SECRET_ID and COS_SECRET_KEY. 4. org.apache.hadoop.fs.auth.SessionTokenCredentialProvider: Accesses by using a temporary key as described in Generating and Using Temporary Keys. 5. org.apache.hadoop.fs.auth.CVMInstanceCredentialsProvider: Gets a temporary key to access COS by using the role bound to CVM. 6. org.apache.hadoop.fs.auth.CPMInstanceCredentialsProvider: Gets a temporary key to access COS by using the role bound to CPM. 	<p>If this parameter follows:</p> <ol style="list-style-type: none"> 1. org.apact 2. org.apact 3. org.apache. 4. org.apact 5. org.apact 6. org.apact 7. org.apact

	<p>7. org.apache.hadoop.fs.auth.EMRInstanceCredentialsProvider: Gets a temporary key to access COS by using the role bound to EMR.</p> <p>8. org.apache.hadoop.fs.auth.RangerCredentialsProvider: Gets a key by using Ranger.</p>	
fs.cosn.useHttps	Whether to use HTTPS as the transfer protocol for the COS backend.	true
fs.cosn.impl	The COSN implementation class for FileSystem, which is fixed at org.apache.hadoop.fs.CosFileSystem.	None
fs.AbstractFileSystem.cosn.impl	The COSN implementation class for AbstractFileSystem, which is fixed at org.apache.hadoop.fs.CosN.	None
fs.cosn.bucket.region	<p>The region of the bucket to access. For enumerated values, see the region abbreviations in Regions and Access Endpoints, such as ap-beijing and ap-guangzhou.</p> <p>This parameter is compatible with the old parameter fs.cosn.userinfo.region.</p>	None
fs.cosn.bucket.endpoint_suffix	The COS endpoint to connect (optional). Public cloud COS users only need to provide the correct region in the parameter above. This parameter is compatible with the old parameter fs.cosn.userinfo.endpoint_suffix. To make endpoint take effect, you should delete the fs.cosn.bucket.region parameter first.	None
fs.cosn.tmp.dir	An existing local directory where temporary files generated at runtime are stored.	/tmp/hadoop
fs.cosn.upload.part.size	<p>The size of each part for multipart upload through the COSN file system. A COS multipart upload supports a maximum of 10,000 parts to be uploaded for a single object. You need to estimate the desired part size as needed.</p> <p>For example, if the part size is set to 8388608 (8 MB), you can upload an object of up to 78 GB in size. The size of a part can be up to 2 GB, that is, the size of a single object can be up to 19 TB.</p>	8388608 (8
fs.cosn.upload.buffer	<p>The type of buffer used when files are uploaded through COSN. Currently, there are three types of buffers: non_direct_memory, direct_memory, and mapped_disk.</p> <p>The non-direct memory buffer uses JVM on-heap memory, the direct_memory buffer uses off-heap memory, and the mapped_disk buffer works based on memory file mapping.</p>	mapped_dis
fs.cosn.upload.buffer.size	The size of buffer used during upload through COSN. A value of -1 means unlimited.	-1

	You can specify this value only if you set the buffer type to mapped_disk. If you specify a value greater than 0, it cannot be smaller than the block size. This parameter is compatible with the old parameter fs.cosn.buffer.size.	
fs.cosn.block.size	The size of a block in the COSN file system.	134217728
fs.cosn.upload_thread_pool	The number of concurrent threads when files are uploaded to COS through streams.	10
fs.cosn.copy_thread_pool	The number of threads used to copy and delete concurrent files during directory replication.	3
fs.cosn.read.ahead.block.size	The size of a read-ahead block.	1048576 (1
fs.cosn.read.ahead.queue.size	The length of the read-ahead queue.	8
fs.cosn.maxRetries	The maximum number of retries if an error occurs when accessing COS.	200
fs.cosn.retry.interval.seconds	The time interval between retries in seconds.	3
fs.cosn.server-side-encryption.algorithm	The COS server-side encryption algorithm. Valid values: SSE-C, SSE-COS. If this parameter is left empty (default value), no encryption algorithm will be used.	None
fs.cosn.server-side-encryption.key	The required SSE-C key if the SSE-C server encryption algorithm is used. This parameter is a Base64-encoded AES-256 key. If this parameter is left empty (default value), no encryption key will be used.	None
fs.cosn.client-side-encryption.enabled	Whether to enable client-side encryption, which is not enabled by default. After enabling it, you must configure the public key and private key for it, and you cannot use the APPEND and TRUNCATE APIs.	false
fs.cosn.client-side-encryption.public.key.path	The absolute path of the public key file for client-side encryption.	None
fs.cosn.client-side-encryption.private.key.path	The absolute path of the private key file for client-side encryption.	None
fs.cosn.crc64.checksum.enabled	Whether to enable CRC-64 checksum. It is disabled by default, meaning that you can't run the <code>hadoop fs -checksum</code> command to	false

	obtain the CRC-64 checksum of a file.	
fs.cosn.crc32c.checksum.enabled	Whether to enable CRC32C checksum. It is disabled by default, meaning that you cannot run the <code>hadoop fs -checksum</code> command to obtain the CRC32C checksum of a file. CRC-64 and CRC32C cannot be both enabled.	false
fs.cosn.traffic.limit	The limit on the upload bandwidth in bits/s. Value range: 819200–838860800. Default value: -1 (unlimited).	None

Hadoop configuration

Modify `$HADOOP_HOME/etc/hadoop/core-site.xml` by adding the information of COS users and implementation classes as shown below:

```
<configuration>
  <property>
    <name>fs.cosn.credentials.provider</name>
    <value>org.apache.hadoop.fs.auth.SimpleCredentialProvider</value>
    <description>
```

This option allows the user to specify how to get the credentials. Comma-separated class names of credential provider classes which implement `com.qcloud.cos.auth.COSCredentialsProvider`:

1. `org.apache.hadoop.fs.auth.SessionCredentialProvider`: Obtain the secret
2. `org.apache.hadoop.fs.auth.SimpleCredentialProvider`: Obtain the secret from `fs.cosn.userinfo.secretId` and `fs.cosn.userinfo.secretKey` in `core-s`
3. `org.apache.hadoop.fs.auth.EnvironmentVariableCredentialProvider`: Obtain from system environment variables named `COS_SECRET_ID` and `COS_SECRET_KE`

If unspecified, the default order of credential providers is:

1. `org.apache.hadoop.fs.auth.SessionCredentialProvider`
2. `org.apache.hadoop.fs.auth.SimpleCredentialProvider`
3. `org.apache.hadoop.fs.auth.EnvironmentVariableCredentialProvider`
4. `org.apache.hadoop.fs.auth.SessionTokenCredentialProvider`
5. `org.apache.hadoop.fs.auth.CVMInstanceCredentialsProvider`
6. `org.apache.hadoop.fs.auth.CPMInstanceCredentialsProvider`
7. `org.apache.hadoop.fs.auth.EMRInstanceCredentialsProvider`

```
</description>
</property>

<property>
  <name>fs.cosn.userinfo.secretId</name>
```

```
<value>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</value>
<description>Tencent Cloud Secret Id</description>
</property>

<property>
  <name>fs.cosn.userinfo.secretKey</name>
  <value>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</value>
  <description>Tencent Cloud Secret Key</description>
</property>

<property>
  <name>fs.cosn.bucket.region</name>
  <value>ap-xxx</value>
  <description>The region where the bucket is located.</description>
</property>

<property>
  <name>fs.cosn.bucket.endpoint_suffix</name>
  <value>cos.ap-xxx.myqcloud.com</value>
  <description>
    COS endpoint to connect to.
    For public cloud users, it is recommended not to set this option, and only
  </description>
</property>

<property>
  <name>fs.cosn.impl</name>
  <value>org.apache.hadoop.fs.CosFileSystem</value>
  <description>The implementation class of the CosN FileSystem.</description>
</property>

<property>
  <name>fs.AbstractFileSystem.cosn.impl</name>
  <value>org.apache.hadoop.fs.CosN</value>
  <description>The implementation class of the CosN AbstractFileSystem.</description>
</property>

<property>
  <name>fs.cosn.tmp.dir</name>
  <value>/tmp/hadoop_cos</value>
  <description>Temporary files will be placed here.</description>
</property>

<property>
  <name>fs.cosn.upload.buffer</name>
  <value>mapped_disk</value>
  <description>The type of upload buffer. Available values: non_direct_memory
```

```
</property>

<property>
  <name>fs.cosn.upload.buffer.size</name>
  <value>134217728</value>
  <description>The total size of the upload buffer pool. -1 means unlimited.</description>
</property>

<property>
  <name>fs.cosn.upload.part.size</name>
  <value>8388608</value>
  <description>Block size to use cosn filesystem, which is the part size for COS. Considering the COS supports up to 10000 blocks, user should estimate the max part size. For example, 8MB part size can allow writing a 78GB single file.</description>
</property>

<property>
  <name>fs.cosn.maxRetries</name>
  <value>3</value>
  <description>The maximum number of retries for reading or writing files to COS, before we signal failure to the application.</description>
</property>

<property>
  <name>fs.cosn.retry.interval.seconds</name>
  <value>3</value>
  <description>The number of seconds to sleep between each COS retry.</description>
</property>

<property>
  <name>fs.cosn.server-side-encryption.algorithm</name>
  <value></value>
  <description>The server-side encryption algorithm.</description>
</property>

<property>
  <name>fs.cosn.server-side-encryption.key</name>
  <value></value>
  <description>The SSE-C server-side encryption key.</description>
</property>

<property>
  <name>fs.cosn.client-side-encryption.enabled</name>
  <value></value>
  <description>Enable or disable the client encryption function</description>
```

```

</property>

<property>
  <name>fs.cosn.client-side-encryption.public.key.path</name>
  <value>/xxx/xxx.key</value>
  <description>The direct path to the public key</description>
</property>

<property>
  <name>fs.cosn.client-side-encryption.private.key.path</name>
  <value>/xxx/xxx.key</value>
  <description>The direct path to the private key</description>
</property>

</configuration>

```

You are not advised to configure `fs.defaultFS` in the production environment. To use it for certain test cases (for example, hive-testbench), add the following configurations:

```

<property>
  <name>fs.defaultFS</name>
  <value>cosn://examplebucket-1250000000</value>
  <description>
    This option is not advice to config, this only used for some special t
  </description>
</property>

```

Server-side encryption

Hadoop-COS supports server-side encryption using either of the following two methods: COS-managed keys (SSE-COS) and customer-provided keys (SSE-C). You can enable this feature, which is disabled by default, by configuring as shown below:

SSE-COS encryption

SSE-COS encryption refers to server-side encryption with a COS-managed key. In this mode, COS manages the master key and its data. When using Hadoop-COS, you can add the following configuration in the

`$HADOOP_HOME/etc/hadoop/core-site.xml` file to implement SSE-COS encryption.

```

<property>
  <name>fs.cosn.server-side-encryption.algorithm</name>
  <value>SSE-COS</value>
  <description>The server-side encryption algorithm.</description>
</property>

```

SSE-C encryption

SSE-C encryption refers to server-side encryption with customer-provided keys. That is, the encryption key is provided by the user. When you upload an object, COS will use the encryption key that you provide to apply AES-256 encryption to the data. When using Hadoop-COS, you can add the following configuration in the `$`

`HADOOP_HOME/etc/hadoop/core-site.xml` file to implement SSE-C encryption.

```
<property>
  <name>fs.cosn.server-side-encryption.algorithm</name>
  <value>SSE-C</value>
  <description>The server-side encryption algorithm.</description>
</property>
<property>
  <name>fs.cosn.server-side-encryption.key</name>
  <value>MDEyMzQ1Njc4OUFCQ0RFRjAxMjMONTY3ODlBQkNERUY=</value> #Users need to
  <description>The SSE-C server-side encryption key.</description>
</property>
```

Note:

The SSE-C encryption feature of Hadoop-COS relies on the SSE-C server-side encryption of COS. This means Hadoop-COS does not store any user-defined encryption keys just like COS. Instead, COS stores HMAC values with random data added to the encryption keys to authenticate access requests. COS cannot use the HMAC values to derive the value of an encryption key or decrypt the content of an object. Therefore, if you lose your encryption key, you will not be able to access the object again.

If you configure an SSE-C server-side encryption algorithm in Hadoop-COS, you must also configure an SSE-C key by using the `fs.cosn.server-side-encryption.key` parameter in the format of a Base64-encoded AES-256 key.

Client-side encryption

COSN client-side encryption adopts the RSA encryption method. The key is divided into public key and private key. The former is used for file encryption, and the latter is used for file decryption. When a file is uploaded, COSN will generate a random key and use it to encrypt the file symmetrically. The public key encrypts this key and saves the encrypted information in the file's metadata. When the file is downloaded, COSN will use the private key to obtain the encrypted random key from the file's metadata for decryption and then use the decrypted random key to decrypt the file. The public and private keys only participate in the local calculation in the client but are not transferred on the network or stored on the server, ensuring the data security of the master key.

When using client-side encryption, you should ensure the integrity and accuracy of the master key. When copying or migrating encrypted data, you should ensure the integrity and accuracy of the cryptographic metadata. If any encrypted data cannot be decoded due to cryptographic metadata loss/corruption caused by your incorrect use or loss of the master key, you shall bear all losses and consequences arising from it.

After client-side encryption is enabled, APPEND and TRUNCATE APIs are no longer supported.

If you run the `hadoop fs -cp` command on an encrypted file in a client with the client-side encryption feature disabled, the encrypted information will be lost.

After client-side encryption is enabled, CRC file verification and async multipart upload are disabled by default.

When using Hadoop-COS, you can add the following configuration in the `$HADOOP_HOME/etc/hadoop/core-site.xml` file to implement SSE-COS encryption.

```
<property>
  <name>fs.cosn.client-side-encryption.enabled</name>
  <value>true</value>
  <description>Enable or disable the client encryption function</description>
</property>

<property>
  <name>fs.cosn.client-side-encryption.public.key.path</name>
  <value>/xxx/xxx.key</value>
  <description>The direct path to the public key</description>
</property>

<property>
  <name>fs.cosn.client-side-encryption.private.key.path</name>
  <value>/xxx/xxx.key</value>
  <description>The direct path to the private key</description>
</property>
```

You can generate the key with the following code:

```
import java.io.FileOutputStream;
import java.io.IOException;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.NoSuchAlgorithmException;
import java.security.PrivateKey;
import java.security.PublicKey;
import java.security.SecureRandom;
import java.security.spec.PKCS8EncodedKeySpec;
import java.security.spec.X509EncodedKeySpec;

// Use asymmetric key RSA encryption for randomly generated symmetric keys
public class BuildKey {
    private static final SecureRandom srand = new SecureRandom();
    private static void buildAndSaveAsymKeyPair(String pubKeyPath, String priKeyPat
        KeyPairGenerator keyGenerator = KeyPairGenerator.getInstance("RSA");
        keyGenerator.initialize(1024, srand);
        KeyPair keyPair = keyGenerator.generateKeyPair();
        PrivateKey privateKey = keyPair.getPrivate();
        PublicKey publicKey = keyPair.getPublic();
```

```

X509EncodedKeySpec x509EncodedKeySpec = new X509EncodedKeySpec(publicKey.getEncoded());
FileOutputStream fos = new FileOutputStream(pubKeyPath);
fos.write(x509EncodedKeySpec.getEncoded());
fos.close();

PKCS8EncodedKeySpec pkcs8EncodedKeySpec = new PKCS8EncodedKeySpec(privateKey.getEncoded());
fos = new FileOutputStream(priKeyPath);
fos.write(pkcs8EncodedKeySpec.getEncoded());
fos.close();
}

public static void main(String[] args) throws Exception {

    String pubKeyPath = "pub.key";
    String priKeyPath = "pri.key";
    buildAndSaveAsymKeyPair(pubKeyPath, priKeyPath);
}
}

```

How to Use

Examples

Run a command in the format of `hadoop fs -ls -R cosn://<BucketName-APPID>/<path>` or `hadoop fs -ls -R /<path>` (you need to set `fs.defaultFS` to `cosn://BucketName-APPID`). The following example uses a bucket named `examplebucket-1250000000`, to which you can append a specific path.

```

hadoop fs -ls -R cosn://examplebucket-1250000000/
-rw-rw-rw-  1 root root      1087 2018-06-11 07:49 cosn://examplebucket-1250000000/
drwxrwxrwx - root root          0 1970-01-01 00:00 cosn://examplebucket-1250000000/
drwxrwxrwx - root root          0 1970-01-01 00:00 cosn://examplebucket-1250000000/
-rw-rw-rw-  1 root root      1087 2018-06-12 03:26 cosn://examplebucket-1250000000/
-rw-rw-rw-  1 root root      2386 2018-06-12 03:26 cosn://examplebucket-1250000000/
drwxrwxrwx - root root          0 1970-01-01 00:00 cosn://examplebucket-1250000000/
-rw-rw-rw-  1 root root      1087 2018-06-11 07:32 cosn://examplebucket-1250000000/
-rw-rw-rw-  1 root root      2386 2018-06-11 07:29 cosn://examplebucket-1250000000/

```

Run the wordcount provided in MapReduce and execute the following command.

Note:

This example uses `hadoop-mapreduce-examples-2.7.2.jar`. To use a different version of the JAR file, modify the version number.

```
bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.2.jar wordcount
```

If the command is successful, the following information will be returned:

```
File System Counters
COSN: Number of bytes read=72
COSN: Number of bytes written=40
COSN: Number of read operations=0
COSN: Number of large read operations=0
COSN: Number of write operations=0
FILE: Number of bytes read=547350
FILE: Number of bytes written=1155616
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=0
HDFS: Number of bytes written=0
HDFS: Number of read operations=0
HDFS: Number of large read operations=0
HDFS: Number of write operations=0
Map-Reduce Framework
Map input records=5
Map output records=7
Map output bytes=59
Map output materialized bytes=70
Input split bytes=99
Combine input records=7
Combine output records=6
Reduce input groups=6
Reduce shuffle bytes=70
Reduce input records=6
Reduce output records=6
Spilled Records=12
Shuffled Maps =1
Failed Shuffles=0
Merged Map outputs=1
GC time elapsed (ms)=0
Total committed heap usage (bytes)=653262848
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
```



```
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=36
File Output Format Counters
Bytes Written=40
```

Accessing COSN through Java code

```
package com.qcloud.chdfs.demo;

import org.apache.commons.io.IOUtils;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FSDataInputStream;
import org.apache.hadoop.fs.FSDataOutputStream;
import org.apache.hadoop.fs.FileChecksum;
import org.apache.hadoop.fs.FileStatus;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;

import java.io.IOException;
import java.net.URI;
import java.nio.ByteBuffer;

public class Demo {
    private static FileSystem initFS() throws IOException {
        Configuration conf = new Configuration();
        // For more information on COSN configuration items, visit https://cloud.tencent.com/document/product/436/125
        // The following configuration items are required
        conf.set("fs.cosn.impl", "org.apache.hadoop.fs.CosFileSystem");
        conf.set("fs.AbstractFileSystem.cosn.impl", "org.apache.hadoop.fs.CosN");
        conf.set("fs.cosn.tmp.dir", "/tmp/hadoop_cos");
        conf.set("fs.cosn.bucket.region", "ap-guangzhou");
        conf.set("fs.cosn.userinfo.secretId", "AKXXXXXXXXXXXXXXXXXXXX");
        conf.set("fs.cosn.userinfo.secretKey", "XXXXXXXXXXXXXXXXXXXX");
        conf.set("fs ofs.user.appid", "XXXXXXXXXXXX");
        // For more information on other configuration items, visit https://cloud.tencent.com/document/product/436/125
        // Whether to enable CRC-64 checksum. It is disabled by default, meaning that the data is not verified.
        conf.set("fs.cosn.crc64.checksum.enabled", "true");
        String cosnUrl = "cosn://f4mxxxxxxxxx-125xxxxxxxx";
        return FileSystem.get(URI.create(cosnUrl), conf);
    }

    private static void mkdir(FileSystem fs, Path filePath) throws IOException {
        fs.mkdirs(filePath);
    }
}
```

```
private static void createFile(FileSystem fs, Path filePath) throws IOException
    // Create a file (if it already exists, it will be overwritten)
    // if the parent dir does not exist, fs will create it!
    FSDataOutputStream out = fs.create(filePath, true);
    try {
        // Write a file
        String content = "test write file";
        out.write(content.getBytes());
    } finally {
        IOUtils.closeQuietly(out);
    }
}

private static void readFile(FileSystem fs, Path filePath) throws IOException {
    FSDataInputStream in = fs.open(filePath);
    try {
        byte[] buf = new byte[4096];
        int readLen = -1;
        do {
            readLen = in.read(buf);
        } while (readLen >= 0);
    } finally {
        IOUtils.closeQuietly(in);
    }
}

private static void queryFileOrDirStatus(FileSystem fs, Path path) throws IOExc
    FileStatus fileStatus = fs.getFileStatus(path);
    if (fileStatus.isDirectory()) {
        System.out.printf("path %s is dir\\n", path);
        return;
    }
    long fileLen = fileStatus.getLen();
    long accessTime = fileStatus.getAccessTime();
    long modifyTime = fileStatus.getModificationTime();
    String owner = fileStatus.getOwner();
    String group = fileStatus.getGroup();

    System.out.printf("path %s is file, fileLen: %d, accessTime: %d, modifyTime
        path, fileLen, accessTime, modifyTime, owner, group);
}

private static void getFileChecksum(FileSystem fs, Path path) throws IOExceptio
    FileChecksum checksum = fs.getFileChecksum(path);
    System.out.printf("path %s, checksumType: %s, checksumCrcVal: %d\\n",
        path, checksum.getAlgorithmName(), ByteBuffer.wrap(checksum.getByte
    }
```

```
private static void copyFileFromLocal(FileSystem fs, Path cosnPath, Path localP
    fs.copyFromLocalFile(localPath, cosnPath);
}

private static void copyFileToLocal(FileSystem fs, Path cosnPath, Path localPat
    fs.copyToLocalFile(cosnPath, localPath);
}

private static void renamePath(FileSystem fs, Path oldPath, Path newPath) throw
    fs.rename(oldPath, newPath);
}

private static void listDirPath(FileSystem fs, Path dirPath) throws IOException
    FileStatus[] dirMemberArray = fs.listStatus(dirPath);

    for (FileStatus dirMember : dirMemberArray) {
        System.out.printf("dirMember path %s, fileLen: %d\\n", dirMember.getPat
    }
}

// The recursive deletion flag is used to delete directories
// If recursion is `false` and `dir` is not empty, the operation will fail
private static void deleteFileOrDir(FileSystem fs, Path path, boolean recursive
    fs.delete(path, recursive);
}

private static void closeFileSystem(FileSystem fs) throws IOException {
    fs.close();
}

public static void main(String[] args) throws IOException {
    // Initialize a file
    FileSystem fs = initFS();

    // Create a file
    Path cosnFilePath = new Path("/folder/exampleobject.txt");
    createFile(fs, cosnFilePath);

    // Read a file
    readFile(fs, cosnFilePath);

    // Query a file or directory
    queryFileOrDirStatus(fs, cosnFilePath);

    // Get a file checksum
    getFileChecksum(fs, cosnFilePath);
}
```

```
// Copy a file from the local system
Path localFilePath = new Path("file:///home/hadoop/ofs_demo/data/exampleobj");
copyFileFromLocal(fs, cosnFilePath, localFilePath);

// Download a file to the local file system
Path localDownFilePath = new Path("file:///home/hadoop/ofs_demo/data/exampleobj");
copyFileToLocal(fs, cosnFilePath, localDownFilePath);

listDirPath(fs, cosnFilePath);
// Rename
mkdir(fs, new Path("/doc"));
Path newPath = new Path("/doc/example.txt");
renamePath(fs, cosnFilePath, newPath);

// Delete a file
deleteFileOrDir(fs, newPath, false);

// Create a directory
Path dirPath = new Path("/folder");
mkdir(fs, dirPath);

// Create a file in a directory
Path subFilePath = new Path("/folder/exampleobject.txt");
createFile(fs, subFilePath);

// List directories
listDirPath(fs, dirPath);

// Delete a directory
deleteFileOrDir(fs, dirPath, true);
deleteFileOrDir(fs, new Path("/doc"), true);

// Close a file system
closeFileSystem(fs);
}
}
```

FAQs

If you have any questions about Hadoop-COS, see [FAQs > Tools > Hadoop](#).

COSDistCp

Last updated : 2024-01-06 16:15:35

Feature Overview

COSDistCp is a MapReduce-based distributed file copy tool mainly used for data copy between HDFS and COS. It introduces the following features:

Performs incremental file migration and data verification based on length and CRC checksum.

Filters files in the source directory with regular expression.

Decompresses files in the source directory and compresses them to the target compression format.

Aggregates text files based on a regular expression.

Preserves user/user group, extension attributes, and time of the source file and directory.

Configures alarms and Prometheus monitoring.

Collects the statistics of file size distribution.

Limits the read bandwidth.

Operating Environments

Operating system

Linux

Software dependency

Hadoop 2.6.0 or above; Hadoop-COS 5.9.3 or above.

Download and Installation

Obtaining the COSDistCp JAR package

If your Hadoop version is 2.x, you can download [cos-distcp-1.13-2.8.5.jar](#) and verify the integrity of the downloaded JAR package according to the MD5 checksum of the package.

If your Hadoop version is 3.x, you can download [cos-distcp-1.13-3.1.0.jar](#) and verify the integrity of the downloaded JAR package according to the MD5 checksum of the package.

Installation notes

In the Hadoop environment, install [Hadoop-COS](#) and then run the COSDistCp tool.

You can download the corresponding versions of COSDistCp, Hadoop-COS, and cos_api-bundle JAR packages from download addresses listed above according to the Hadoop version. Then, specify the Hadoop-COS-related parameters to perform the copy operation, where the JAR package addresses should be the local addresses:

```
hadoop jar cos-distcp-${version}.jar \\  
-libjars cos_api-bundle-${version}.jar,hadoop-cos-${version}.jar \\  
-Dfs.cosn.credentials.provider=org.apache.hadoop.fs.auth.SimpleCredentialProvider \\  
-Dfs.cosn.userinfo.secretId=COS_SECRETID \\  
-Dfs.cosn.userinfo.secretKey=COS_SECRETKEY \\  
-Dfs.cosn.bucket.region=ap-guangzhou \\  
-Dfs.cosn.impl=org.apache.hadoop.fs.CosFileSystem \\  
-Dfs.AbstractFileSystem.cosn.impl=org.apache.hadoop.fs.CosN \\  
--src /data/warehouse \\  
--dest cosn://examplebucket-1250000000/warehouse
```

How It Works

COSDistCp uses the MapReduce framework. The multi-process and multi-thread tool performs operations such as file copy, data verification, compression, file attribute preservation, and copy retries. COSDistCp will overwrite files with the same name in the destination location. If data copy or verification fails, the corresponding file may fail to be copied and information about these files will be written in a temporary directory. If new files are added to your source file system or the file content changes, you can use the `--skipMode` or `--diffMode` parameter to compare the length or CRC checksum of the files to implement data verification and incremental file migration.

Parameter description

You can run the `hadoop jar cos-distcp-${version}.jar --help` (`${version}` is the version number) command under the `hadoop` user to view the COSDistCp-supported parameters. The following table describes the parameters of the COSDistCp of the current version:

Attribute Key	Description	Default Value
<code>--help</code>	Outputs parameters supported by COSDistCp. Example: <code>--help</code>	None
<code>--src=LOCATION</code>	Location of the data to copy. This can be either an HDFS or COS location. Example: <code>--src=hdfs://user/logs/</code>	None
<code>--dest=LOCATION</code>	Destination for the data. This can be either an	None

	HDFS or COS location. Example: --dest=cosn://examplebucket-1250000000/user/logs	
--srcPattern=PATTERN	A regular expression that filters files in the source location. Example: --srcPattern='.*\\.log\$' *Note: Enclose your parameter in single quotation marks (') in case asterisks (*) are parsed by the shell.**	None
--taskNumber=VALUE	Number of copy threads Example: --taskNumber=10	10
--workerNumber=VALUE	Number of copy threads. COSDistCp will create a copy thread pool for each copy process based on this value set. Example: workerNumber=4	4
--filesPerMapper=VALUE	The number of files input to each mapper. Example: --filesPerMapper=10000	500000
--groupBy=PATTERN	A regular expression to concatenate text files that match the regular expression. Example: --groupBy='.*group-input/(\\d+)-(\\d+).*'	None
--targetSize=VALUE	The size (in MB) of the files to create. This parameter is used together with --groupBy. Example: --targetSize=10	None
--outputCodec=VALUE	Compression method of output file. Valid values: gzip, lzo, snappy, none, keep. Here: 1. keep indicates to keep the compression method of the original file. 2. none indicates to decompress the file based on the file extension. Example: --outputCodec=gzip Note: if the /dir/test.gzip and /dir/test.gz files exist, and you specify the output format as lzo, only /dir/test.lzo will be retained.	keep
--deleteOnSuccess	Deletes the source file immediately after it is successfully copied to the destination directory. Example: --deleteOnSuccess Note: v1.7 and later no longer provide this	false

	parameter. We recommend you delete the data in the source file system after migrating the data successfully and using --diffMode for verification.	
--multipartUploadChunkSize=VALUE	The size (in MB) of the multipart upload part transferred to COS using the Hadoop-COS plugin. COS supports up to 10,000 parts. You can set the value based on the file size. Example: --multipartUploadChunkSize=20	8 MB
--cosServerSideEncryption	Specifies whether to use SSE-COS for encryption on the COS server side. Example: --cosServerSideEncryption	false
--outputManifest=VALUE	Creates a file (Gzip compressed) that contains a list of all files copied to the destination location. Example: --outputManifest=manifest.gz	None
--requirePreviousManifest	If this parameter is set to true, --previousManifest=VALUE must be specified for incremental copy. Example: --requirePreviousManifest	false
--previousManifest=LOCATION	A manifest file that was created during the previous copy operation. Example: --previousManifest=cosn://examplebucket-1250000000/big-data/manifest.gz	None
--copyFromManifest	Copies files specified in --previousManifest to the destination file system. This is used together with previousManifest=LOCATION. Example: --copyFromManifest	false
--storageClass=VALUE	The storage class to use. Valid values: STANDARD, STANDARD_IA, ARCHIVE, DEEP_ARCHIVE, and INTELLIGENT_TIERING. For more information, see Storage Class Overview .	None
--srcPrefixesFile=LOCATION	A local file that contains a list of source directories, one directory per line. Example: --srcPrefixesFile=file:///data/migrate-folders.txt	None

--skipMode=MODE	Verifies whether the source and destination files are the same before the copy. If they are the same, the file will be skipped. Valid values are none (no verification), length, checksum, length-mtime, and length-checksum. Example: --skipMode=length	length-checksum
--checkMode=MODE	Verifies whether the source and destination files are the same when the copy is completed. Valid values are none (no verification), length, checksum, length-mtime, and length-checksum. Example: --checkMode=length-checksum	length-checksum
--diffMode=MODE	Specifies the rule for obtaining the list of different files in the source and destination directories. Valid values are length, checksum, length-mtime, and length-checksum. Example: --diffMode=length-checksum	None
--diffOutput=LOCATION	Specifies the HDFS output directory in diffMode. This directory must be empty. Example: --diffOutput=/diff-output	None
--cosChecksumType=TYPE	Specifies the CRC algorithm used by the Hadoop-COS plugin. Valid values are CRC32C and CRC64. Example: --cosChecksumType=CRC32C	CRC32C
--preserveStatus=VALUE	Specifies whether to copy the user, group, permission, xattr, and timestamps metadata of the source file to the destination file. Valid values are any combinations of letters u, g, p, x, and t (initials of user, group, permission, xattr, and timestamps, respectively). Example: --preserveStatus=ugpt	None
--ignoreSrcMiss	Ignores files that exist in the manifest file but cannot be found during the copy.	false
--promGatewayAddress=VALUE	Specifies the Prometheus PushGateway address and port for pushing the counter data of MapReduce jobs.	None
--promGatewayDeleteOnFinish=VALUE	Whether to delete JobName metrics from Prometheus PushGateway when the specified	true

	job is completed. Example: --promGatewayDeleteOnFinish=true	
--promGatewayJobName=VALUE	JobName to report to Prometheus PushGateway Example: --promGatewayJobName=cos-distcp-hive-backup	None
--promCollectInterval=VALUE	Interval to collect MapReduce jobs, in ms Example: --promCollectInterval=5000	5000
--promPort=VALUE	Server port to expose Prometheus metrics Example: --promPort=9028	None
--enableDynamicStrategy	Enables the dynamic task assignment policy to make tasks with quicker migration migrate more files. Note: This mode has certain limits; for example, the task counter may be inaccurate if the process is abnormal. Therefore, use --diffMode to verify the data after migration. Example: --enableDynamicStrategy	false
--splitRatio=VALUE	Split ratio of the dynamic strategy. A higher splitRatio indicates a smaller job granularity. Example: --splitRatio=8	8
--localTemp=VALUE	Local folder to store the job files generated by the dynamic strategy Example: --localTemp=/tmp	/tmp
--taskFilesCopyThreadNum=VALUE	Number of concurrency to copy the job files generated by the dynamic strategy to the HDFS Example: --taskFilesCopyThreadNum=32	32
--statsRange=VALUE	Statistics range Example: --- statsRange=0,1mb,10mb,100mb,1gb,10gb,inf	0,1mb,10mb,100mb,
--printStatsOnly	Collects only statistics on the file size distribution without copying the data. Example: --printStatsOnly	None
--bandWidth	Maximum bandwidth for reading each migrated file (in MB/s). Default value: -1, which	None

	indicates no limit on the read bandwidth. Example: --bandWidth=10	
--jobName	Migration task name. Example: --jobName=cosdistcp-to-warehouse	None
--compareWithCompatibleSuffix	Whether to change the source file extension gzip to gz and lzop to lzo when using the --skipMode and --diffMode parameters. Example: --compareWithCompatibleSuffix	None
--delete	Moves files that exist in the source directory but not in the target directory to the separate trash directory and generates the file list in order to ensure the file consistency between the source and target directories. Note: This parameter cannot be used together with --diffMode.	None
--deleteOutput	Specifies the HDFS output directory for delete. This directory must be empty. Example: --deleteOutput=/dele-output	None

Example

Viewing the help option

Run the following command with `--help` to view the parameters supported by COSDistCp:

```
hadoop jar cos-distcp-${version}.jar --help
```

In the command above, `${version}` is the version ID of the COSDistCp. For example, the name of the COSDistCp JAR package (version 1.0) is `cos-distcp-1.0.jar`.

File size distribution of the files to copy

Run the following command with the `--printStatsOnly` and `--statsRange=VALUE` parameters to output the file size distribution of the files to copy:

```
hadoop jar cos-distcp-${version}.jar --src /wookie/data --dest cosn://examplebucket

Copy File Distribution Statistics:
Total File Count: 4
Total File Size: 1190133760
| SizeRange          | TotalCount          | TotalSize          |
```

0MB ~ 1MB	0 (0.00%)	0 (0.00%)	
1MB ~ 10MB	1 (25.00%)	1048576 (0.09%)	
10MB ~ 100MB	1 (25.00%)	10485760 (0.88%)	
100MB ~ 1024MB	1 (25.00%)	104857600 (8.81%)	
1024MB ~ 10240MB	1 (25.00%)	1073741824 (90.22%)	
10240MB ~ LONG_MAX	0 (0.00%)	0 (0.00%)	

Specifying the source and destination locations for the files to copy

Run the following command with the `--src` and `--dest` parameters:

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://examplebucket
```

COSDistCp will retry 5 times for files that failed to be copied. If the copy still fails, these files will be written to the `/tmp/${randomUUID}/output/failed/` directory, where `${randomUUID}` is a random string. After recording the failed file information, COSDistCp will continue to migrate the remaining files, and the migration task will not fail due to the migration failure of some files. When the migration task is completed, COSDistCp will output counter information (ensure that your task submitting machine is configured with INFO log output for MapReduce jobs on the submission end) and determine whether there are files that failed to be migrated, and if yes, it will throw an exception on the client that submitted the task.

The following information about a source file might be contained in the output:

1. SRC_MISS: The copy fails because the source file contained in the manifest is not found.
2. COPY_FAILED: The copy fails due to other reasons.

You can run the copy command again to implement incremental migration. Run the following command to obtain the log of the MapReduce job. In this way, you can find out the cause of the copy failure. Note that

`application_1610615435237_0021` is the application ID.

```
yarn logs -applicationId application_1610615435237_0021 > application_1610615435237
```

Querying counters

When the copy operation ends, statistics on the copy will be output. The counters are as follows:

```
CosDistCp Counters
  BYTES_EXPECTED=10198247
  BYTES_SKIPPED=10196880
  FILES_COPIED=1
  FILES_EXPECTED=7
  FILES_FAILED=1
  FILES_SKIPPED=5
```

The statistics are described as follows:

Statistics Item	Description
-----------------	-------------

BYTES_EXPECTED	Total size (in bytes) to copy according to the source directory
FILES_EXPECTED	Number of files to copy according to the source directory, including the directory itself
BYTES_SKIPPED	Total size (in bytes) of files that can be skipped (same length or checksum value)
FILES_SKIPPED	Number of source files that can be skipped (same length or checksum value)
FILES_COPIED	Number of source files that are successfully copied
FILES_FAILED	Number of source files that failed to be copied
FOLDERS_COPIED	Number of directories that are successfully copied
FOLDERS_SKIPPED	Number of directories that are skipped

Specifying the number of copy processes and the number of threads in each process

Run the following command with the `--taskNumber` and `--workersNumber` parameters. COSDistCp adopts a multi-process, multi-thread framework for the copy operation. You can:

Use `--taskNumber` to specify the number of processes.

Use `--workerNumber` to specify the number of threads in each copy process.

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse/ --dest cosn://examplebu
```

Skipping files with the same check value for incremental migration

Run the following command with the `--skipMode` parameter to skip copying source files with the same length and checksum as those of destination files. The default value is `length-checksum` :

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://examplebu
```

`--skipMode` is used to verify whether the source and destination files are the same before the copy. If they are the same, the file will be skipped. Valid values are `none` (no verification), `length` , `checksum` , and `length-checksum` (length + CRC checksum).

If the checksum algorithms of the source and destination file systems are different, the source file will be read for calculating a new checksum. If your source is HDFS, you can identify whether the HDFS source supports the COMPOSITE-CRC32C algorithm as follows:

```
hadoop fs -Ddfs.checksum.combine.mode=COMPOSITE_CRC -checksum /data/test.txt
/data/test.txt COMPOSITE-CRC32C 6a732798
```

Verifying data after migration and migrating incremental data

Run the command with the `--diffMode` and `--diffOutput` parameters:

`--diffMode` can be set to `length` or `length-checksum`.

`--diffMode=length` obtains the list of different files based on whether the file sizes are the same.

`--diffMode=length-checksum` obtains the list of different files based on whether the file size and CRC checksum are the same.

`--diffOutput` specifies the output directory for the diff operation.

If the destination file system is COS and the CRC algorithm of the source file system is different from that of COS, COSDistCp will pull the source file to calculate the CRC checksum of the destination file system and compare the CRC checksums to check whether they are the same. In the following sample code, the `--diffMode` parameter is used to check whether the source and destination files are the same based on the file size and CRC checksum after migration.

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://examplebucket
```

After the above command is executed successfully, the counter information based on the file list of the source file system will be output (ensure that your task submitting machine is configured with INFO log output for MapReduce jobs on the submission end). You can analyze whether the source and destination files are the same based on the counter information as detailed below:

1. SUCCESS: the source and destination files are the same.
2. DEST_MISS: The destination file does not exist.
3. SRC_MISS: The source file contained in the source file manifest is not found during the verification.
4. LENGTH_DIFF: Sizes of the source and destination files are different.
5. CHECKSUM_DIFF: CRC checksums of the source and destination files are different.
6. DIFF_FAILED: The `diff` operation fails due to insufficient permissions or other reasons.
7. TYPE_DIFF: the source is a directory but the destination is a file.

In addition, COSDistCp will generate a list of different files in the `/tmp/diff-output/failed` directory in HDFS (or `/tmp/diff-output` for v1.0.5 or earlier versions). You can run the following command to obtain the list of different files except for those recorded as SRC_MISS:

```
hadoop fs -getmerge /tmp/diff-output/failed diff-manifest
grep -v '"comment":"SRC_MISS"' diff-manifest |gzip > diff-manifest.gz
```

Run the following command to implement incremental copy based on the list of different files:

```
hadoop jar cos-distcp-${version}.jar --taskNumber=20 --src /data/warehouse --dest
```

After incremental migration is completed, run the command with the `--diffMode` parameter again to check whether the files are completely identical.

Verifying whether the source and destination files have the same CRC checksum

Run the command with the `--checkMode` parameter to check whether the source and destination files have the same length and checksum after file copy is completed. The default value is `length-checksum`.

When you are copying files from a non-COS file system to COS, if the CRC algorithms of the source and Hadoop-COS are different, the CRC checksum will be calculated during the copy operation. When the copy operation is completed, the CRC checksum of the destination file will be obtained and compared with the calculated CRC checksum of the source file.

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://exampleb
```

Note:

It takes effect if `--groupBy` is not specified and `--outputCodec` is the default value.

Restricting the read bandwidth for a single file

Run the command with the `--bandWidth` parameter (in MB). The following example command restricts the read bandwidth of each copied file to 10 MB/s:

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://examplebu
```

Copying multiple directories

You can create a local file (for example, `srcPrefixes.txt`) and add the absolute paths of multiple directories to copy to the file (the directories cannot be in parent-child relationships). After this, you can run the `cat` command to view the directories as follows:

```
cat srcPrefixes.txt
/data/warehouse/20181121/
/data/warehouse/20181122/
```

Then, you can use `--srcPrefixesFile` to specify this file. The command is as follows:

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --srcPrefixesFile file
```

Filtering source files with a regular expression

Run the following command with the `--srcPattern` parameter. In this example, only files whose extension is ".log" in the `/data/warehouse/` directory are copied.

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse/ --dest cosn://exampleb
```

Do not copy files whose extension is ".temp" or ".tmp":

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse/ --dest
cosn://examplebucket-1250000000/data/warehouse/ --srcPattern='.*(?
```

```
<!\.temp|\.tmp)$'
```

Specifying the checksum type of Hadoop-COS

Run the following command with the `--cosChecksumType` parameter. Valid values are `CRC32C` (default) and `CRC64`.

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://examplebu
```

Specifying the storage class for COS objects

Run the following command with the `--storageClass` parameter:

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://examplebuc
```

Specifying the output compression codec

Run the command with the `--outputCodec` parameter, which allows you to compress HDFS data to COS in real time to reduce storage costs. Valid values are `keep`, `none`, `gzip`, `lzop`, and `snappy`. If the parameter is set to `none`, the files will be copied uncompressed. If it is set to `keep`, the files will be copied with no change in their compression. The following is an example:

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse/logs --dest cosn://examp
```

Note:

If the parameter is not set to `keep`, the files will be decompressed and converted to the target compression format. Due to the difference in compression parameters, the content of the destination files might be different from that of the source files, but the files will be the same after decompression. If `--groupBy` is not specified and `--outputCodec` is the default value, you can use `--skipMode` to perform incremental migration and `--checkMode` to perform data verification.

Deleting the source files

Run the command with the `--deleteOnSuccess` parameter. The following example deletes the corresponding source files in the `/data/warehouse` directory immediately after they are copied from HDFS to COS:

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://examplebuc
```

Note:

If `--deleteOnSuccess` is specified, each source file is deleted immediately after the file is copied, but not after all source files are copied. The parameter is not provided in version 1.7 or later.

Generating the target manifest and specifying the previous manifest

Run the command with the `--outputManifest` and `--previousManifest` parameters.

`--outputManifest` generates a local `manifest.gz` (Gzip compressed) file. When the copy operation is successful, the file is moved to the directory specified in `--dest`.

`--previousManifest` specifies the destination files that are copied during the previous copy operation (`--outputManifest`). COSDistCp will skip files of the same size.

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://examplebu
```

Note:

The command above performs incremental copy only. Only files with size changes can be copied. If the file content is changed, you can refer to the example of `--diffMode` and determine the changed manifest files based on the CRC checksum.

Using dynamic strategy for migration job distribution

If your files differ greatly in size, (e.g., there are a few large files, causing imbalanced loads of a large number of small files and machines), you can use `--enableDynamicStrategy` to enable the dynamic strategy, which allows faster-speed jobs to copy more files to speed up the whole copy process.

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://exampl
```

Verify the migrated data after migration is completed:

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest  
cosn://examplebucket-1250000000/data/warehouse/ --diffMode=length-checksum --  
diffOutput=/tmp/diff-output
```

Note:

This mode has certain limits; for example, the task counter may be inaccurate if the process is exceptional. Therefore, use `--diffMode` to verify the data after migration.

Copying metadata of the source file

Run the following command with the `--preserveStatus` parameter to copy the `user`, `group`, `permission`, and `timestamps` (modification time and access time) metadata of the source file/directory to the destination file/directory. The parameter takes effect when files are copied from HDFS to CHDFS.

Sample:

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://examplebuc
```

Configuring Prometheus

You can go to YARN to view the COSDistCp job counter, including the number of files/bytes that have been copied. To easily view the graph of the COSDistCp jobs, you can display the data using Prometheus and Grafana with easy

configurations. The following example configures `prometheus.yml` to add the jobs to grab:

```
- job_name: 'cos-distcp-hive-backup'
  static_configs:
    - targets: ['172.16.16.139:9028']
```

Run the command with the `--promPort=VALUE` parameter to expose the counter of the current MapReduce job:

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest cosn://examplebu
```

Download the sample [Grafana Dashboard](#) and import it. The Grafana dashboard will be as follows:



Alarms for copy failures

Run the command with the `--completionCallbackClass` parameter to specify the path of the callback class.

When the task is completed, COSDistCp will use the collected task information as parameters to execute the callback function. For user-defined callback functions, the following APIs need to be implemented. You can download the [callback sample code](#).

```
package com.qcloud.cos.distcp;
import java.util.Map;
public interface TaskCompletionCallback {
/**
 * @description: When the task is completed, the callback function is executed
 * @param jobType Copy or Diff
 * @param jobStartTime the job start time
 * @param errorMsg the exception error msg
 * @param applicationId the MapReduce application id
 * @param cosDistCpCounters the job
 */
}
```

```
void doTaskCompletionCallback(String jobType, long jobStartTime, String errorMsg, S

/**
 * @description: init callback config before execute
 */
void init() throws Exception;
}
```

COSDistCp has integrated the alarms of Cloud Monitor. When the task runs abnormally or some files fail to be copied, alarming will be performed.

```
export alarmSecretId=SECRET-ID
export alarmSecretKey=SECRET-KEY
export alarmRegion=ap-guangzhou
export alarmModule=module
export alarmPolicyId=cm-xxx
hadoop jar cos-distcp-1.4-2.8.5.jar \
-Dfs.cosn.credentials.provider=org.apache.hadoop.fs.auth.SimpleCredentialProvider \
-Dfs.cosn.userinfo.secretId=SECRET-ID \
-Dfs.cosn.userinfo.secretKey=SECRET-KEY \
-Dfs.cosn.bucket.region=ap-guangzhou \
-Dfs.cosn.impl=org.apache.hadoop.fs.CosFileSystem \
-Dfs.AbstractFileSystem.cosn.impl=org.apache.hadoop.fs.CosN \
--src /data/warehouse \
--dest cosn://examplebucket-1250000000/data/warehouse/ \
--checkMode=checksum \
--completionCallbackClass=com.qcloud.cos.distcp.DefaultTaskCompletionCallback
```

`alarmPolicyId` in the command above is an alarm policy created in Cloud Monitor. You can go to the Cloud Monitor console (**Alarm Management > Alarm Configuration > Custom Messages**) to create and configure one.

FAQs

What stages are involved in migration of HDFS data with COSDistCp? How do I adjust the migration performance and ensure the data correctness?

COSDistCp verifies each migrated file upon migration completion according to `checkMode` :

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest
cosn://examplebucket-1250000000/data/warehouse --taskNumber=20
```

After migration is completed, you can also run the following command to view the list of different source and destination files:

```
hadoop jar cos-distcp-${version}.jar --src /data/warehouse --dest  
cosn://examplebucket-1250000000/data/warehouse/ --diffMode=length-checksum --  
diffOutput=/tmp/diff-output
```

How can I run COSDistCp if Hadoop-COS is not configured in the environment?

You can download a specific version of the COSDistCp JAR package according to the Hadoop version and specify the Hadoop-COS-related parameters to perform the copy operation.

```
hadoop jar cos-distcp-${version}.jar \<\  
-Dfs.cosn.credentials.provider=org.apache.hadoop.fs.auth.SimpleCredentialProvider \<\  
-Dfs.cosn.userinfo.secretId=COS_SECRETID \<\  
-Dfs.cosn.userinfo.secretKey=COS_SECRETKEY \<\  
-Dfs.cosn.bucket.region=ap-guangzhou \<\  
-Dfs.cosn.impl=org.apache.hadoop.fs.CosFileSystem \<\  
-Dfs.AbstractFileSystem.cosn.impl=org.apache.hadoop.fs.CosN \<\  
--src /data/warehouse \<\  
--dest cosn://examplebucket-1250000000/warehouse
```

What should I do if the result shows that some files failed to be copied?

COSDistCp will retry 5 times for IOException occurred during the copy process. If the copy still fails, information about the failed files will be written to the `/tmp/${randomUUID}/output/failed/` directory, where

`${randomUUID}` is a random string. Common reasons for the copy failure are as follows:

1. The source file contained in the copy manifest is not found during the copy (recorded as SRC_MISS).
2. The job initiator does not have permission to read the source file or write the destination file, or the copy failed due to other reasons (recorded as COPY_FAILED).

If the log message indicates that the source file does not exist, and the source file is ignorable, you can run the following command to obtain the list of different files except for those recorded as SRC_MISS:

```
hadoop fs -getmerge /tmp/${randomUUID}/output/failed/ failed-manifest  
grep -v '"comment":"SRC_MISS"' failed-manifest |gzip > failed-manifest.gz
```

Except for those recorded as SRC_MISS, if there are other failed files, you can locate the failure reasons by referring to the error log messages in the `/tmp/${randomUUID}/output/logs/` directory and pulling the application logs. The following command example pulls the logs of the YARN application:

```
yarn logs -applicationId application_1610615435237_0021 > application_1610615435237
```

In the command above, `application_1610615435237_0021` is the application ID.

Will COSDistCp generate incomplete files due to network or other exceptions?

If the network is abnormal, the source file is missing, or the permissions are insufficient, COSDistCp cannot generate a destination file with the same size as the source file.

For versions earlier than COSDistCp 1.5, COSDistCp will attempt to delete the destination files generated. If the deletion fails, you need to re-execute the copy task to overwrite the incomplete files, or manually delete them.

If your COSDistCp version is 1.5 or later and the version of the Hadoop-COS plugin in the running environment is 5.9.3 or later, when files fail to be copied to COS, COSDistCp will call the abort API to terminate the ongoing upload request. Therefore, no incomplete file will be generated even if an exception occurs.

If your COSDistCp version is 1.5 or later but the version of Hadoop-COS in the running environment is earlier than 5.9.3, you are advised to upgrade it to 5.9.3 or later.

If the destination location is not COS, COSDistCp will attempt to delete the destination files.

There are some invisible incomplete multipart uploads in COS buckets, which occupy storage space. How do I deal with them?

COS buckets may have some incomplete multipart uploads occupying storage space due to incidents such as server exception and process kill. You can configure an incomplete multipart upload deletion rule as instructed in [Setting Lifecycle](#) to clear them.

A memory overflow and task timeout occurred during migration. How do I adjust parameters?

During migration, both COSDistCp and the tools used to access COS and CHDFS, based on their own logic, occupy some memory. To avoid memory overflow and task timeout, you can adjust parameters of some MapReduce jobs, for example:

```
hadoop jar cos-distcp-${version}.jar -Dmapreduce.task.timeout=18000 -  
Dmapreduce.reduce.memory.mb=8192 --src /data/warehouse --dest  
cosn://examplebucket-1250000000/data/warehouse
```

As shown in the example above, the value of `mapreduce.task.timeout` is changed to 18,000 seconds to avoid job timeout when large files are copied, and the value of `mapreduce.reduce.memory.mb` (memory size of the Reduce process) is changed to 8 GB to avoid memory overflow.

How do I control the migration bandwidth of the migration task through migration over Direct Connect?

The formula for calculating the total bandwidth limit of COSDistCp migration is: `taskNumber` x `workerNumber` x `bandWidth`. You can set `workerNumber` to 1, use the `taskNumber` parameter to control the number of concurrent migrations, and use the `bandWidth` parameter to control the bandwidth of a single concurrent migration.

HDFS TO COS

Last updated : 2024-01-06 16:15:35

Overview

The HDFS TO COS tool is used to copy data from HDFS to Tencent Cloud COS (Cloud Object Storage) .

Operating Environments

Operating system

Linux or Windows

Software dependency

JDK v1.7 or v1.8

Installation and configuration

For more information on environment installation and configuration, see [Java](#).

Configuration Method

1. Install Hadoop v2.7.2 or higher. For detailed directions, see [Hadoop](#).
2. Download the HDFS TO COS tool from [GitHub](#) and decompress it.
3. Copy the `core-site.xml` file of the HDFS cluster to be synced to the `conf` folder. The `core-site.xml` file contains the configuration information of NameNode.
4. In the `cos_info.conf` configuration file, configure the bucket, region, and API key information. The bucket name is formed by connecting a user-defined string and the system-generated `APPID` with a hyphen, for example, `examplebucket-1250000000` .
5. Specify the configuration file location in the command line parameter. The default location is `conf/cos_info.conf` .

Note:

If the command line parameter conflicts with the configuration file, the command line parameter shall apply.

Usage

Note:

Linux is used as an example below.

Viewing help

```
./hdfs_to_cos_cmd -h
```

Copying a file

Copy from HDFS to COS. If a file with the same name as the file to be copied already exists in COS, the former will be overwritten.

```
./hdfs_to_cos_cmd --hdfs_path=/tmp/hive --cos_path=/hdfs/20170224/
```

Copy from HDFS to COS. If a file with the same name and length as the file to be copied already exists in COS, the latter will be skipped (this is suitable for repeated copy).

```
./hdfs_to_cos_cmd --hdfs_path=/tmp/hive --cos_path=/hdfs/20170224/ -  
skip_if_len_match
```

Only the length is checked here, as the overheads would be very high if the digests of files in Hadoop are calculated.

Copy from HDFS to COS. If the `Har` directory (Hadoop archive file) exists in HDFS, the `.har` files can be automatically decompressed by specifying the `--decompress_har` parameter:

```
./hdfs_to_cos_cmd --decompress_har --hdfs_path=/tmp/hive --  
cos_path=/hdfs/20170224/
```

If the `--decompress_har` parameter is not specified, the directory will be copied as an ordinary HDFS directory, that is, the files in the `Har` directory such as `index` and `masterindex` will be copied as-is.

Directory information

```
conf: configuration file, which is used to store `core-site.xml` and `cos_info.conf`  
log: log directory  
src: Java source program  
dep: compiled executable JAR package
```

FAQs and Help

Configuration information

Please make sure that the entered configuration information is correct, including bucket, region, and API key information. The bucket name is formed by connecting the user-defined string and system-generated `APPID` with a hyphen, such as `examplebucket-1250000000`. Please also make sure that the time on the server is in sync with the local time (if there is a difference of about 1 minute, it is okay, but if the difference is large, please set the server time correctly).

DataNode

Please make sure that the server where the copy program is located can also access the DataNode. The NameNode uses a public IP address and can be accessed, but the DataNode where the obtained block is located uses a private IP address and cannot be accessed directly; therefore, it is recommended that the copy program be placed in a Hadoop node for execution, so that both the NameNode and DataNode can be accessed.

Permissions

Please use the current account to download a file with the Hadoop command, check whether everything is correct, and then use the synchronization tool to sync the data in Hadoop.

File overwriting

Files that already exist in COS will be overwritten by default in case of repeated upload, unless you explicitly specify the `-skip_if_len_match` parameter, which indicates to skip files during upload if they have the same length as existing files.

cos path

`cos path` is considered as a directory by default, and files that are eventually copied from HDFS will be stored in this directory.

Copying data from Tencent Cloud EMR HDFS

To copy data from Tencent Cloud EMR HDFS to COS, you are advised to use the high-performance DistCp tool. For more information, see [Migrating Data Between HDFS and COS](#).

GooseFS-Lite

Last updated : 2025-01-07 15:55:28

Feature Description

The GooseFS-Lite tool supports mounting [Cloud Object Storage \(COS\)](#) buckets locally, allowing you to directly operate objects in Tencent Cloud Object Storage as if using a local file system. Compared to the COSFS tool, GooseFS-Lite offers higher large file read and write speeds and is not limited by local disk performance. GooseFS-Lite supports major POSIX file system features, such as file order, random read, sequential write, and directory manipulation.

Use Limits

GooseFS-Lite is only suitable for simple management of files after mounting and does not support some features of local file systems. Please note the following usage limits:

Files cannot be written randomly and truncate operations are not supported.

When multiple clients mount the same COS bucket, users need to self-coordinate the behavior of multiple clients, such as avoiding multiple clients writing to the same file.

Rename operations for files/folders are non-atomic.

Reading and renaming files being written in the current mount point are not supported.

Metadata operations like listing directories have poor performance due to the need for remote access to the COS server.

Soft/hard links are not supported.

Append write performance is poor, involving server-side data copy and downloading the appended file.

Not recommended for use in low memory scenarios, such as when container memory or CVM memory is less than 2G.

Container environments are currently only supported within [Tencent Kubernetes Engine \(TKE\)](#). Non-TKE containers are not supported.

Not recommended for use in scenarios with a large amount of random reads and high performance requirements.

Note:

External network mount and append write operations on non-infrequent storage will incur download traffic fees.

Usage Environment

[KonaJDK 11](#)

Linux X86_64

Usage

Step 1. Install dependencies

CentOS/TencentOS Server

```
yum install -y fuse-devel
```

Ubuntu

```
apt install -y libfuse-dev
```

Other Linux distributions

Compile and install libfuse2.9.7

```
wget "https://github.com/libfuse/libfuse/releases/download/fuse-2.9.7/fuse-2.9.7.tar.gz"
tar xvf fuse-2.9.7.tar.gz
cd fuse-2.9.7
./configure
make -j8
make install
```

Step 2: Installing GooseFS-Lite

Install GooseFS-Lite in the current directory and soft link `goosefs-lite` to `/usr/bin/goosefs-lite` for convenient use of the `goosefs-lite` command.

```
curl -fssL https://downloads.tencentgoosefs.cn/goosefs-lite/install.sh | sh -x
cd goosefs-lite-*
sudo bash bin/install.sh
```

Step 3: Installing KonaJDK11

In the `goosefs-lite-<specific version>` directory (for example, in the `goosefs-lite-1.0.6` directory), use the following command to install KonaJDK to `/usr/local/konajdk11`:

```
sudo bash bin/install-jdk.sh https://github.com/Tencent/TencentKona-11/releases/download/kona11.0.22/TencentKona-11.0.22.b1-jdk_linux-x86_64.tar.gz
```

As shown below, there are two options to choose from:

First: Use the download link for konajdk. Second: Download the konajdk installation package to the specified directory. Use the following command to install it, so that goosefs-lite can automatically use this Java runtime environment.

```
Usage:
Command: install-jdk.sh http[s]://host/path
Example: install-jdk.sh https://github.com/Tencent/TencentKona-11/releases/download
or
Command: install-jdk.sh /path/to/jdk.tar.gz
Example: install-jdk.sh /Downloads/TencentKona-11.0.22.b1-jdk_linux-x86_64.tar.gz
```

If you want a more flexible way to install the Java environment, refer to [manual JDK installation](#) and modify environment variables in **conf/goosefs-env.sh** to make it effective.

Step 4: Modifying the Configuration File

In the `goosefs-lite-<specific version>` directory (for example, in the `goosefs-lite-1.0.6` directory), there are two ways to modify the configuration file:

Use sed to modify the following three parameters, filling in SECRET_ID, SECRET_KEY, and REGION as needed:

Set fs.cosn.userinfo.secretKey to the Tencent Cloud key.

Set fs.cosn.userinfo.secretId to the Tencent Cloud ID.

Set fs.cosn.bucket.region to the bucket region.

```
sed -i '/<name>fs.cosn.userinfo.secretId<\/name>/{N;s/<value>[^<]*<\/value>/<value>SECRET_ID/g}'
sed -i '/<name>fs.cosn.userinfo.secretKey<\/name>/{N;s/<value>[^<]*<\/value>/<value>SECRET_KEY/g}'
sed -i '/<name>fs.cosn.bucket.region<\/name>/{N;s/<value>[^<]*<\/value>/<value>REGION/g}'
```

You can also use vim to edit the conf/core-site.xml file to modify parameters.

Set fs.cosn.userinfo.secretKey to the Tencent Cloud key.

Set fs.cosn.userinfo.secretId to the Tencent Cloud ID.

Set fs.cosn.bucket.region to the bucket region.

Configuration file description

In the `goosefs-lite-<specific version>/conf` directory (for example, in the `goosefs-lite-1.0.6/conf` directory), you can see the following files:

acl-site.properties: specify directory permissions, username, group name, similar to Linux Posix semantics 0755, uid, gid, etc.

core-site.xml: Hadoop-COS configuration file (goosefs-lite's data flow is based on Hadoop-COS, so parameters and configuration files are almost universal)

goosefs-env.sh: various environment variables, such as JVM parameters, etc.

goosefs-lite.properties: goosefs-lite configuration.

log4j.properties: log configuration. To enable debug logging, uncomment the last line of this file and remount.

acl-site.properties: specify directory permissions, username, group name, similar to Linux Posix semantics 0755, uid, gid, etc.

Note:

We recommend that users avoid using permanent keys in the configuration. Configuring sub-account keys or temporary keys can help improve business security. When authorizing a sub-account, grant only the permissions of the operations and resources that the sub-account needs, which helps avoid unexpected data leakage.

If you must use a permanent key, we recommend that you limit ITS permission scope. This can improve the security by limiting its executable operations, resource scope, and conditions (access IP, etc.).

Step 5: Mounting a Bucket to a Local Directory

Execute the following command in the `goosefs-lite-<specific version>` directory (for example, in the `goosefs-lite-1.0.6` directory) to mount the bucket configured in the key file to the specified directory:

```
./bin/goosefs-lite mount <MountPoint> cosn://<BucketName>/
```

Among them:

<MountPoint> is the local mounting directory (for example, `/mnt/goosefs-lite-mnt-dir`), which must be empty; otherwise, it cannot be mounted.

<BucketName> is the bucket name (e.g., `examplebucket-1250000000`).

Example:

```
mkdir -p /mnt/goosefs-lite-mnt
./bin/goosefs-lite mount /mnt/goosefs-lite-mnt/ cosn://examplebucket-1250000000/
```

View the local mount point and the corresponding COS bucket. The output information includes the process ID, local mount point, and COS path in sequence:

```
$ ./bin/goosefs-lite stat
pid      mount_point          cos_path
13815    /mnt/goosefs-lite-mnt/ cosn://examplebucket-1250000000/
```

If you need to specify multiple mount parameters simultaneously in the command line, you can use commas to separate multiple parameters. The following command sets the mount point to read-only and allows other users to access the mount point:

```
./bin/goosefs-lite mount -o "ro,allow_other" mnt/ cosn://examplebucket-1250000000/
```

Among them:

-o allow_other: To allow other users to access the mount folder, you can specify this parameter when running GooseFS-Lite.

-o ro: Set the mount point to read-only, disallowing write and delete operations.

Note:

A single parameter can be specified with `-o` , such as `-o ro` ; multiple parameters can be separated by commas, such as `-o "ro,allow_other"` .

Step 6: Uninstalling the Mount Point

In the `goosefs-lite-<specific version>` directory (for example, in the `goosefs-lite-1.0.6` directory), execute the following command to uninstall the mount point:

```

$ ./bin/goosefs-lite umount /mnt/goosefs-lite-mnt
Unmount fuse at /mnt/goosefs-lite-mnt/ (PID: 17206).

# If uninstallation is abnormal, you can force uninstall using the following command
$ sudo umount -l /mnt/goosefs-lite-mnt

```

Step 7: Parameter Tuning

GooseFS-Lite includes two configuration files: `conf/core-site.xml` and `conf/goosefs-lite.properties`.

You can optimize upload and download bandwidth by modifying `conf/core-site.xml`. Common parameters are as follows, more parameters can be found in the [Hadoop-COS](#) documentation.

Attribute Key	Description	Default Value	Required
<code>fs.cosn.useHttps</code>	Configure whether to use HTTPS as the transfer protocol for the COS backend.	true	No
<code>fs.cosn.upload.part.size</code>	The size of each part in the chunked upload. Since COS can only support up to 10,000 parts in a chunked upload, it is necessary to estimate the maximum possible single file size. For example, when the part size is 8MB, a single file with a maximum size of 78GB can be uploaded. The maximum part size can be supported is 2GB, which means a single file can support up to 19TB.	8388608 (8MB)	No
<code>fs.cosn.upload_thread_pool</code>	The number of concurrent threads when files are uploaded to COS through streams.	32	No
<code>fs.cosn.read.ahead.block.size</code>	The size of the pre-read block.	1048576 (1MB)	No
<code>fs.cosn.read.ahead.queue.size</code>	The length of the read-ahead queue.	6	No

fs.cosn.trsf.fs ofs.tmp.cache.dir	Temporary file directory of Metadata Accelerator bucket .	No	Metadata Accelerator bucket is required
fs.cosn.trsf.fs ofs.user.appid	Appid of Metadata Accelerator bucket .	No	Metadata Accelerator bucket is required
fs.cosn.trsf.fs ofs.bucket.region	Region of Metadata Accelerator bucket , such as ap-shanghai, ap-beijing.	No	Metadata Accelerator bucket is required

You can adjust the behavior of GooseFS-Lite by modifying conf/goosefs-lite.properties. Common parameters are as follows:

Attribute	Description	Default Value	Required
goosefs.fuse.list.entries.cache.enabled	Enable client List cache	true	No
goosefs.fuse.list.entries.cache.max.size	Maximum number of entries in client List cache, unit: entries	100000	No
goosefs.fuse.list.entries.cache.max.expiration.time	Valid time of client List cache, unit: ms	15000	No
goosefs.fuse.async.release.max.wait.time	Time to wait for write operation to complete when files in open and rename operations are being written, unit: ms	5000	No
goosefs.fuse.umount.timeout	Time to wait for incomplete operations when uninstalling the file system, unit: ms	120000	No

When your read and write concurrency is high, you can adjust the maximum JVM runtime memory of GooseFS-Lite as follows to avoid FullGC and OutOfMemoryError. The default JVM values are `-Xmx512m -XX:MaxDirectMemorySize=512m -XX:+UseG1GC -XX:G1HeapRegionSize=32m`. The adjustment method is as follows:

```
export JAVA_OPTS=" -Xms2G -Xmx2G"
```

```
./bin/goosefs-lite mount /mnt/goosefs-lite-mnt/ cosn://examplebucket-12500000000/  
ps -ef|grep goosefs-lite|grep -v grep
```

FAQs

Missing libfuse library file, how to handle it?

```
java.lang.UnsatisfiedLinkError: /tmp/libjnfuse2359305659114946750.so: libfuse.so.2: cannot open shared object file: No such file or directory  
at java.base/java.lang.ClassLoader$NativeLibrary.load0(Native Method)  
at java.base/java.lang.ClassLoader$NativeLibrary.load(ClassLoader.java:2445)  
at java.base/java.lang.ClassLoader$NativeLibrary.loadLibrary(ClassLoader.java:2501)  
at java.base/java.lang.ClassLoader.loadLibrary0(ClassLoader.java:2700)  
at java.base/java.lang.ClassLoader.loadLibrary(ClassLoader.java:2630)  
at java.base/java.lang.Runtime.load0(Runtime.java:768)  
at java.base/java.lang.System.load(System.java:1837)  
at com.qcloud.cos.goosefs.jnifuse.utils.NativeLibraryLoader.loadLibraryFromJar(NativeLibraryLoader.java:105)  
at com.qcloud.cos.goosefs.jnifuse.utils.NativeLibraryLoader.loadLibrary(NativeLibraryLoader.java:83)  
at com.qcloud.cos.goosefs.jnifuse.LibFuse.loadLibrary(LibFuse.java:48)  
at com.qcloud.cos.goosefs.jnifuse.LibFuse.<clinit>(LibFuse.java:32)  
at com.qcloud.cos.goosefs.jnifuse.AbstractFuseFileSystem.<clinit>(AbstractFuseFileSystem.java:41)  
at com.qcloud.cos.goosefs.fuse.GooseFSLiteFuse.main(GooseFSLiteFuse.java:64)  
at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)  
at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)  
at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)  
at java.base/java.lang.reflect.Method.invoke(Method.java:566)  
at org.springframework.boot.loader.MainMethodRunner.run(MainMethodRunner.java:53)  
at java.base/java.lang.Thread.run(Thread.java:829)
```

To install libfuse, follow these steps:

Method 1

1. Run the following command to install fuse-devel.

If you are using CentOS or TencentOS systems, run the following command:

```
yum install fuse-devel
```

If you are using Ubuntu systems, run the following command:

```
apt install libfuse-dev
```

2. After installation, run the following command to check if the installation was successful.

```
find / -name libfuse.so*
```

Method 2

Update the old version libfuse.so.2.9.2. The installation steps are as follows:

Note:

CentOS 7 installs libfuse.so.2.9.2 by default.

1. Download the [libfuse source code](#) and compile and generate libfuse.so.2.9.7.

```
tar -zxvf fuse-2.9.7.tar.gz
cd fuse-2.9.7/ && ./configure && make && make install
echo -e '\\n/usr/local/lib' >> /etc/ld.so.conf
ldconfig
```

2. After compiling and generating libfuse.so.2.9.7, follow these steps to replace it:

2.1 Run the following command to find the old version libfuse.so.2.9.2 library links.

```
find / -name libfuse.so*
```

2.2 Run the following commands to copy libfuse.so.2.9.7 to the location of the old version library libfuse.so.2.9.2.

```
cp /usr/local/lib/libfuse.so.2.9.7 /usr/lib64/
```

2.3 Run the following commands to delete all links of the old version libfuse.so library.

```
rm -f /usr/lib64/libfuse.so
rm -f /usr/lib64/libfuse.so.2
```

2.4 Run the following commands to build libfuse.so.2.9.7 library links similar to those of the old version library.

```
ln -s /usr/lib64/libfuse.so.2.9.7 /usr/lib64/libfuse.so
ln -s /usr/lib64/libfuse.so.2.9.7 /usr/lib64/libfuse.so.2
```

Configuring Boot Mount

1. Edit the file `/usr/lib/systemd/system/goosefs-lite.service` and add the following content. You can replace `examplebucket-1250000000` with your bucket: Note that the memory values configured in `JAVA_OPTS` (`-Xms` and `-Xmx`) should not exceed 50% of the physical memory limit of the node. For example, if the node has 16GB of physical memory, it is recommended to configure up to `-Xms8G -Xmx8G`.

The following uses version `goosefs-lite-1.0.6` as an example:

```
[Unit]
Description=The Tencent Cloud GooseFS Lite for COS
Requires=network-online.target
After=network-online.target

[Service]
Type=forking
User=root
Environment="JAVA_OPTS=-Xms2G -Xmx4G -XX:MaxDirectMemorySize=1G -XX:+UseG1GC -XX:G1HeapRegionSize=32m"
```



```
ExecStart=/usr/local/goosefs-lite-1.0.6/bin/goosefs-lite mount /mnt/goosefs-mnt
cosn://examplebucket-1250000000/
ExecStop=/usr/local/goosefs-lite-1.0.6/bin/goosefs-lite umount /mnt/goosefs-mnt
Restart=always
RestartSec=5

[Install]
WantedBy=multi-user.target
```

2. Run the following command to execute the mount command and view the status of the background Daemon process.

```
# Apply the systemd configuration of goosefs-lite
systemctl daemon-reload
# Start the background Fuse process
systemctl start goosefs-lite
# View the status of the background Daemon process
systemctl status goosefs-lite
# View the mount point list
/usr/local/goosefs-lite-1.0.6/bin/goosefs-lite stat
# If modifying the systemd configuration, reload and restart after
modification.
```

Set to attempt mounting at boot startup:

```
systemctl enable goosefs-lite
```

3. Uninstall the mount point, reboot the machine, and check the status of the Fuse process.

```
# Execute Uninstall. Note: Do not Uninstall during data writing, as it may
cause data incomplete.
systemctl stop goosefs-lite
# Restart operating system. Proceed with caution to avoid affecting business.
reboot -h now
# View the status of the background Daemon process
systemctl status goosefs-lite
# View the mount point list
/usr/local/goosefs-lite-1.0.6/bin/goosefs-lite stat
```

GooseFS-Lite Having High CPU Utilization and Sending a Large Number of HEAD and LIST Requests to COS, Incurring a Large Amount of Request Fees During a Certain Period Every Day, What Should I Do

This is usually caused by a scheduled disk scan task on your machine. The common disk scan program on Linux systems is updatedb. You can add the GooseFS-Lite mount point directory to the PRUNEPATHS configuration item in the updatedb configuration file `/etc/updatedb.conf` to avoid the disk scan behavior of this program. Additionally, you can use the Linux tool auditd to find the programs accessing the GooseFS-Lite mount point.

Directions are as follows:**1. Install auditd.**

If you are using Ubuntu systems, run the following command:

```
apt-get install auditd -y
```

If it is a CentOS system, execute the following command:

```
yum install audit audit-libs
```

2. Start the auditd service.

```
systemctl start auditd  
systemctl enable auditd
```

3. Monitor the mounted directory.**Note:**

`-w` specifies the GooseFS-Lite mounted directory, `-k` indicates the key to be outputted to the audit logs.

```
auditctl -w /usr/local/service/mnt/ -k goosefs_lite_mnt
```

4. Determine the access program based on the logs.

The audit log directory: `/var/log/audit`, the query command is as follows:

```
ausearch -i|grep 'goosefs_lite_mnt'
```

5. Stop auditd service.

If you need to stop the auditd service, you can use the following command:

```
/sbin/service auditd stop
```

Note:

If the program accessing the mount point is always running, the newly started auditd will not monitor the access behavior of the program. This is because only the first call in multiple calls from the program to the mounted directory will be recorded.

Handling "Cannot Allocate Memory" Error During GooseFS-Lite Installation

```
OpenJDK 64-Bit Server VM warning: INFO: os::commit_memory(0x0000000080000000, 2147483648, 0) failed; error='Cannot allocate memory' (errno=12)
#
# There is insufficient memory for the Java Runtime Environment to continue.
# Native memory allocation (mmap) failed to map 2147483648 bytes for committing reserved memory
# An error report file with more information is saved as:
# /usr/goosefs-lite/goosefs-lite-1.0.2/hs_err_pid2860293.log
^
```

This error mainly occurs because of a memory allocation error during the GooseFS-Lite operation, usually when the requested memory exceeds the actual available memory.

You can modify the JAVA_OPT parameter in the ./bin/goosefs-lite file to a reasonable memory value, ensuring that the requested memory amount is less than the instance available memory amount.

Viewing the Latest Package Version

Run the following command, the return value is the latest version number.

```
curl -fsSL https://downloads.tencentgoosefs.cn/goosefs-lite/LATEST_VERSION
```

Viewing Logs

Problem troubleshooting relies on logs. Below are the locations of the relevant logs.

For **goosefs-lite version 1.0.3** and later, the default log directory is under `/data/goosefs/logs/fuse`.

For example: If the user mounting goosefs-lite is root and the mount point path is `/data1/data2`, then the log path is: `/data/goosefs/logs/fuse/root/data1/data2`.

For **goosefs-lite versions 1.0.0-1.0.2**, the default log directory is under `/data/goosefs/logs`.

Another way to check the mount point log path is to first remount, then `ps aux | grep ${MOUNT_POINT}`.

From the output, you can see Error_File or goosefs.logs.dir, and the parent directory of this path is where all logs for the mount point are located.

System logs: If it is a centos or tlinux system, it is `/var/log/message*`. If it is ubuntu, it is `/var/log/syslog`.

If you need to enable debug logs, go to `conf/log4j.properties`, comment out the last line, and remount to take effect.

Note:

Enabling debug logs will impact performance. Normally, there is no need to enable them.

Mount failed with error "Name or service not known", how to handle it?

```
2023-05-16 19:23:20,668 [Thread-8] ERROR jnifuse.AbstractFuseFileSystem (AbstractFuseFileSystem.java:statfsCallback)
Failed to statfs /:
java.lang.RuntimeException: java.net.UnknownHostException: VM-36-104-centos: VM-36-104-centos: Name or service not known
    at com.qcloud.cos.goosefs.util.network.NetworkAddressUtils.getLocalIpAddress(NetworkAddressUtils.java:522)
    at com.qcloud.cos.goosefs.util.network.NetworkAddressUtils.getLocalHostName(NetworkAddressUtils.java:444)
    at com.qcloud.cos.goosefs.util.network.NetworkAddressUtils.getLocalHostMetricName(NetworkAddressUtils.java:463)
    at com.qcloud.cos.goosefs.metrics.MetricsSystem.constructSourceName(MetricsSystem.java:202)
    at com.qcloud.cos.goosefs.metrics.MetricsSystem.lambda$static$0(MetricsSystem.java:89)
    at com.qcloud.cos.goosefs.util.CommonUtils$2.firstTime(CommonUtils.java:799)
    at com.qcloud.cos.goosefs.util.CommonUtils$2.get(CommonUtils.java:794)
    at com.qcloud.cos.goosefs.metrics.MetricsSystem.getMetricNameWithUniqueId(MetricsSystem.java:390)
    at com.qcloud.cos.goosefs.metrics.MetricsSystem.lambda$getClientMetricName$5(MetricsSystem.java:341)
    at java.util.concurrent.ConcurrentHashMap.computeIfAbsent(ConcurrentHashMap.java:1660)
    at com.qcloud.cos.goosefs.metrics.MetricsSystem.getClientMetricName(MetricsSystem.java:340)
    at com.qcloud.cos.goosefs.metrics.MetricsSystem.getMetricName(MetricsSystem.java:271)
    at com.qcloud.cos.goosefs.metrics.MetricsSystem.timer(MetricsSystem.java:529)
    at com.qcloud.cos.goosefs.fuse.GooseFSFuseUtils.call(GooseFSFuseUtils.java:287)
    at com.qcloud.cos.goosefs.fuse.GooseFSLiteJniFuseFileSystem.statfs(GooseFSLiteJniFuseFileSystem.java:1091)
    at com.qcloud.cos.goosefs.jnifuse.AbstractFuseFileSystem.statfsCallback(AbstractFuseFileSystem.java:289)
```

This is usually because the domain name cannot be resolved. You can try pinging the domain name. If the error shown above occurs, you can execute the following command:

```
ping VM-36-104-centos
```

If it also returns a failure, you can configure the corresponding IP by modifying `/etc/hosts`. Generally, you can set it to `127.0.0.1`.

Follow the steps below:

1. Add a line in the `/etc/hosts` file. Replace VM-36-104-centos with your hostname.

```
127.0.0.1 VM-36-104-centos
```

2. Then retest with ping. After confirming normal resolution, remount to take effect.

Default Environment JDK Is Not KonaJDK11, How to Use GooseFS-lite

1. Download the [KonaJDK11](#) package and extract it.
2. Copy the absolute path of the java binary program in konajdk. For example, the extracted jdk is under

```
/root/konajdk11
```

```
# Determine Java Version
/root/konajdk11/bin/java -v
```

```
# Modify the Current Shell's Java Environment Variable Without Affecting Other Shells and Processes
export JAVA=/root/konajdk11/bin/java
# Mounting
goosefs-lite mount /mnt cosn://bucket-appid
```

3. Using `ps aux | grep goosefs-lite`, you can see the process starts with

`/root/konajdk11/bin/java`, indicating that the specified java version is being used, operation completed.

The Mount Point Was Originally Normal, Suddenly Unable to Use During Operation, How to Handle

Assume the current problematic mount point is `/tmp/mount_point`.

1. First, use `ps aux | grep /tmp/mount_point` to check if any process is using this mount point, including `goosefs-lite`. If there is, use the `kill` command to terminate the corresponding process.
2. Use `ls` to attempt to access the mount point. If the return is empty, it means the mount point has been successfully unmounted. Then remount to take effect.

```
ls /mount_point
```

3. If an error like **transport is not connected** is thrown, execute `umount -l /mount_point` to force unmount. (This command requires root privileges.)

Usually, such situations are caused by processes being killed by `kill -9` or the system oom-killer. You can find it in the system logs or `goosefs-lite` logs.

4. Check the mount point log directory (`/data/goosefs/logs/fuse/$USER/$MOUNT_POINT`) for logs starting with `hs_error` (file describes the stack and reason before program exit).

Throwing Exception: Unsupported or Unrecognized SSL Message, how to handle it?

The current environment does not support disabling https mode, configuration needs to be modified. The solution is to add the following content to the `core-site.xml` configuration file:

```
<property>
  <name>fs.con.useHttps</name>
  <value>false</value>
</property>
```

How to Access COS Using Intranet Domain Name in GooseFS-Lite

In the `core-site.xml` configuration file, delete the `fs.cosn.bucket.region` property and add `fs.cosn.bucket.endpoint_suffix` .

To learn more about parameters, read the following documents:

[GitHub - tencentyun/hadoop-cos](#)

[Object Storage Hadoop Tools](#)

Handling 403 Forbidden During Mounting or Usage

Typically, ERROR logs in the log will describe what permissions are missing, supplement as needed.

Note: For head bucket permissions, it needs to be set at the bucket level rather than just the path level. For example,

`cosn://bucket-appid/path` will not work, it must be `cosn://bucket-appid` . This permission will not expose the objects in the bucket.

Handling Error "fuse: failed to open /dev/fuse Operation not permitted"

1. Check if you have root privileges.
2. If in a container, check if the container is started with `--privileged`. If not, add it.
3. Check if the fuse kernel module is installed: execute the command `lsmod | grep fuse` and see if there is any output. If not, it indicates that the fuse kernel module is missing.

Handling File Write Failure with Error "part num: 10001, the parameter partNumber is not valid"

This indicates that the number of parts for multipart upload exceeds the limit. COS supports a maximum of 10,000 parts. Therefore, `goosefs-lite` by default supports files up to 8MB*10000 (approximately 78GB). If you need to support larger files, you need to adjust the parameter `fs.cosn.upload.part.size`. For example, if `fs.cosn.upload.part.size` is changed to 16777216 (i.e., 16MB), it can support large files up to 16MB*10000.

Online Auxiliary Tools

COS Request Tool

Last updated : 2024-01-06 16:15:35

Feature Overview

The COS request tool is a web-based debugging tool provided by COS. It is integrated on the TencentCloud API 3.0 Explorer platform for API debugging.

Note:

Requests sent by the COS request tool will be sent to the real COS server. **As all operations are real, be careful when performing operations such as `DELETE` .**

The COS request tool supports XML APIs but not JSON APIs.

JSON APIs were provided by COS for you to access COS before XML APIs were released. Both types of APIs have the same underlying architecture where data is interconnected; however, they are incompatible with each other.

XML APIs have a richer set of features and strengths over JSON APIs. We strongly recommend you upgrade to XML APIs for COS.

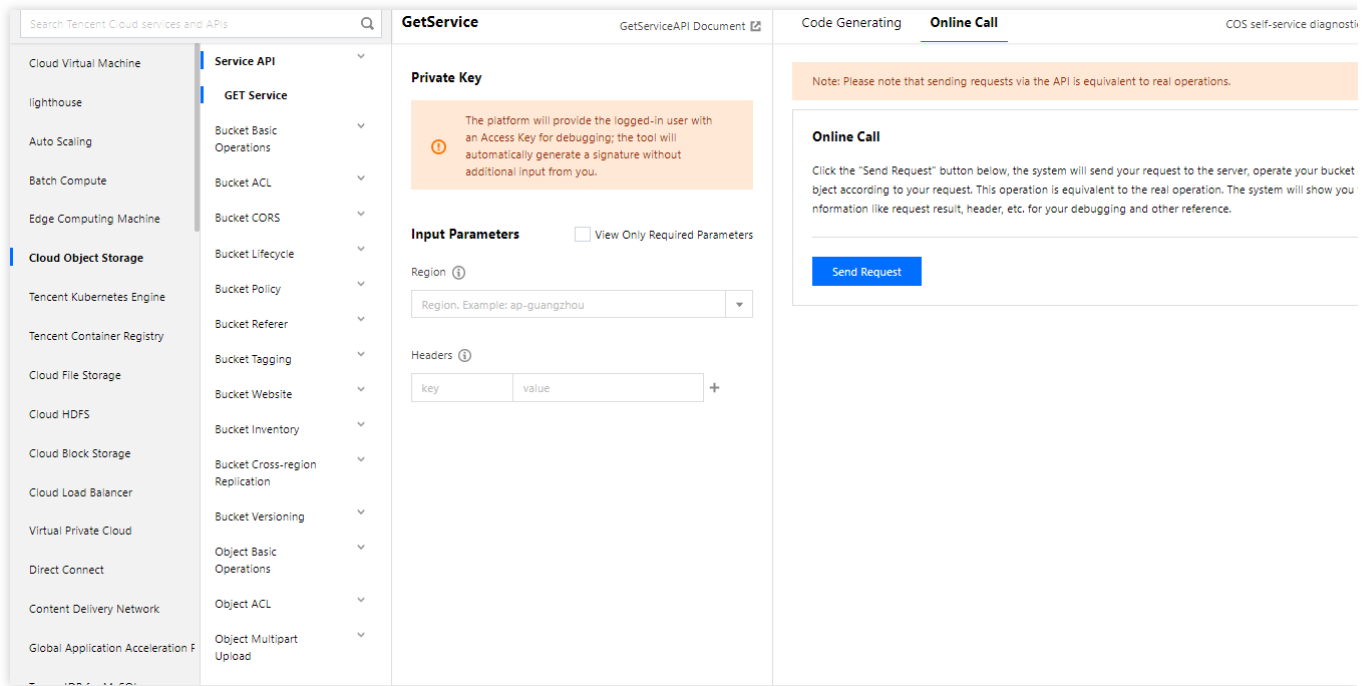
Tool URL

Click [here](#) to enter the COS request tool.

Directions

Select the **Cloud Object Storage** product, select the required API, enter parameters for the API, and click **Send Request** to get the corresponding response.

The COS request tool page shows the sections of product, API, parameter, and result from left to right. You can perform operations in different sections and send the request in the result section to get the response and process parameters.



The detailed steps to use the COS request tool are as shown below:

1. Select the COS product.

Click **Cloud Object Storage** in the product section on the far left, and then you can see COS APIs in the API section.

Note:

The COS request tool is integrated on the TencentCloud API 3.0 platform that provides API debugging tools for many Tencent Cloud products. You can also select other products to debug their APIs as needed.

2. Select the API to be debugged.

You can select the API for debugging as needed. Three types of COS APIs are shown in the API section: service APIs, bucket APIs, and object APIs.

Take `GET Service` as an example for service APIs. This API can list the information of all buckets under your account. Your API key is required. To get the bucket information of your account in a specified region, select the corresponding region in the parameter section. For more information on this API, see [GET Service \(List Buckets\)](#).

Bucket APIs are used to manipulate buckets, such as `PUT Bucket lifecycle`. For more information, see [Bucket APIs](#).

Object APIs are used to manipulate objects, such as `PUT Object`. For more information, see [Object APIs](#).

3. Enter parameters for the API.

The parameter section lists the corresponding parameters for the selected API. For more information on the parameters of COS APIs, see [API Documentation](#).

API key is a required parameter for API calling. When using an API to manipulate resources such as buckets or objects, you need to enter your API key to authorize the API request, which can be obtained on the [Manage API Key](#) page in the CAM console.

Note:

For each API, the COS request tool displays a red asterisk behind each required parameter. You can also select **Only Required Parameters** to view required parameters only in the parameter section.

4. Send a request and view the response.

After selecting an API and entering parameters, click **Send Request** on the **Online Call** tab. Your request will be sent to the server, and the server will manipulate your buckets or objects accordingly.

Note:

Requests sent by the COS request tool will be sent to the real COS server. **As all operations are real, be careful when performing operations such as `DELETE`.**

After the request is sent, the returned result and the request parameters will be displayed in the result section. The **Request Parameters** part lists your HTTP request body; the **Response Result** part lists the response body of the request; the **Signature Process** part lists the signature involved in the request and its generation process; and the **Curl** part lists the statement called by Curl.

Sample

For example, a `GET Object` request is sent to get a file named `0001.txt` as shown below. The **Request Parameters** part lists the corresponding parameters of the request.

```
GET https://bucketname-appid.cos.ap-region.myqcloud.com/0001.txt
Host: bucketname-appid.cos.ap-region.myqcloud.com
Authorization: q-sign-algorithm=sha1&q-ak=AKIDwqaGoCIWIG4hDWdJUTL5e3hn04xi****&q-si
```

The first line shows your HTTP Verb and the link to be accessed; the second line shows the domain name to be accessed; and the last line shows the signature information of the request. For requests of the `PUT` type, request headers are complicated, but there are some common request headers. For more information, see [Common Request Headers](#).

The **Signature Process** part shows the signature involved in this request and its generation process. For more information on signature algorithms, see [Request Signature](#). If you need to generate and debug request signatures, we recommend you use the COS signature tool.

The response result returned by COS is as follows:

```
200 OK
content-type: text/plain
content-length: 6
connection: close
accept-ranges: bytes
date: Wed, 28 Nov 2018 09:42:49 GMT
etag: "5a8dd3ad0756a93ded72b823b19dd877"
last-modified: Tue, 27 Nov 2018 20:05:26 GMT
server: tencent-cos
x-cos-request-id: NWJmZTYzMTlfOWUxYzBiMDlfOTA4NF8yMWI2****
x-cos-version-id: MTg0NDY3NDI1MzAzODkyMjU****
hello!
```

The `200 OK` in the first line is the status code returned for the request. If the request fails, the corresponding error code will be returned. For more information, see [Error Codes](#). Other lines are response headers, which vary by API, but there are some common response headers. For more information, see [Common Response Headers](#).

Notes

After you click **Send Request** to send the request with its required parameters entered to the COS server, COS will perform the corresponding operation on your buckets or objects. The operation cannot be undone or reverted; therefore, proceed with caution.

Diagnostic Tool

Last updated : 2024-01-06 16:15:35

Overview

COS's web-based diagnostic tool allows you to troubleshoot error requests. You can enter the error `RequestId` (see [Obtaining RequestId](#)) on the tool page and click **Diagnose**. The tool will check the request and provide basic information about the request as well as suggestions so that you can quickly troubleshoot COS API errors.

Tool URL

[Diagnosis Tool](#)

Directions

1. Click [Diagnosis Tool](#).
2. Enter the `RequestId` and click **Diagnose**.
3. Wait and view the diagnostic result.

The result includes the suggestions and the request information.

The suggestions help you quickly locate the COS API errors.

The request information is the information about the request corresponding to `RequestId`.

4. Send your feedback on the suggestions.

Click **Helpful** or **Not Helpful** below the suggestions so that we can further improve the tool.

FAQs

On the diagnostic tool page, you can also find the FAQs. If you have any queries, please [contact us](#).

Notes

A COS `RequestId` must:

1. Start with N.
2. Contain at least 30 characters.

3. Comply with the Base64 standard.