Object Storage Service

Utilities

Utilities

ossutil

Download and install the tool

Ossutil allows you to manage OSS data easily using the command line. The current version does not provide complete bucket management and multipart management functions. These functions will be available in subsequent versions. If you need these functions, you can use the osscmd command line tool instead.

Download the tool

Current version

- Current version: 1.0.0

Runtime environment

- Windows/Linux/Mac
- Supported architecture
    - x86 (32bit, 64bit)

Download the binary program

- [Linux x86 32bit] ossutil32
- [Linux x86 64bit] ossutil64
- [Windows x86 32bit] ossutil32.zip
- [Windows x86 64bit] ossutil64.zip
- [mac x86 64bit] ossutilmac64

Install and use the binary program

Download the binary program or corresponding compressed package for your operating system and run the binary program. (If the binary program is not an executable file, run chmod 755 ossutil to make it executable.) That is:

For a Linux system:

./ossutil

For a Window system, either of the following two methods can be used (64-bit operating system as an example):

1) Decompress the package, double-click the bat file, and enter ossutil64.exe.

2) Decompress the package, run cmd to enter the directory where the binary program resides, and enter ossutil64.exe.

For a MAC system:

./ossutilmac64

Quick start

Set ossutil language

When running commands of ossutil, you can use the -L option to set the language. The value can be CH or EN, that is, Chinese or English. The value is case insensitive. The default value is CH (Chinese). If you set the language to CH (Chinese), you must ensure that your system is UTF-8 encoded. Otherwise, garbled characters may be displayed.

For example:

./ossutil help ls is used to display the ls help in the default language.

./ossutil help ls -L ch is used to display the ls help in Chinese.

./ossutil help ls -L en is used to display the ls help in English.

./ossutil config -L ch is used to run an interactive configuration command of ossutil config. The prompt language is Chinese.

./ossutil config -L en is used to run an interactive configuration command of ossutil config. The prompt language is English.

> **Note:** Errors output by ossutil are in English by default, which will not be affected by the preceding options.

Obtain the command list

./ossutil or ./ossutil help

```
$./ossutil
Usage: ossutil [command] [args...] [options...]
Run ossutil help to display the command help.

Commands:
mb cloud_url [options]
Creates a bucket.
ls [cloud_url] [options]
Lists buckets or objects.
rm cloud_url [options]
Deletes a bucket or object.
stat cloud_url [options]
Displays the description of a bucket or object.
set-acl cloud_url [acl] [options]
Sets the ACL for a bucket or object.
set-meta cloud_url [meta] [options]
Sets the meta information of the uploaded objects.
cp src_url dest_url [options]
Uploads, downloads, or copies objects.
restore cloud_url [options]
Restores an object from the frozen state to the readable state.
create-symlink cloud_url target_url [options]
Creates a symbolic link.
read-symlink cloud_url [options]
Reads the description of a symbolic link file.

Additional Commands:
help [command]
Obtains the help document of a command.
config [options]
Creates a configuration file to store configuration items.
hash file_url [options]
Computes the crc64 or MD5 of a local file.
update [options]
Updates ossutil.



$./ossutil  -L en
Usage: ossutil [command] [args...] [options...]
Please use 'ossutil help command' to show help of command

Commands:
mb cloud_url [options]
Make Bucket
ls [cloud_url] [options]
List Buckets or Objects
rm cloud_url [options]
Remove Bucket or Objects
stat cloud_url [options]
Display meta information of bucket or objects
set-acl cloud_url [acl] [options]
Set acl on bucket or objects
set-meta cloud_url [meta] [options]
set metadata on already uploaded objects
cp src_url dest_url [options]
Upload, Download or Copy Objects
```

```
restore cloud_url [options]
Restore Frozen State Object to Read Ready Status
create-symlink cloud_url target_url [options]
Create symlink of object
read-symlink cloud_url [options]
Display meta information of symlink object

Additional Commands:
help [command]
Get help about commands
config [options]
Create configuration file to store credentials
hash file_url [options]
Get crc64 or md5 of local file
update [options]
Update ossutil
```

View the help document of a command

./ossutil help cmdYou are strongly advised to run the help command to view the help document before running a command.

```
./ossutil  help config -L ch
SYNOPSIS

Creates a configuration file to store configuration items.

SYNTAX

ossutil config [-e endpoint] [-i id] [-k key] [-t token] [-L language] [--output-dir outdir] [-c file]

DETAIL DESCRIPTION

This command is used to create a configuration file, store customized configuration items in the configuration file, and provide access information when the OSS is accessed using the configuration items. (Whether a command requires configuration items depends on whether it supports the --config-file option. For details, refer to the command help.)

You can specify the path for storing the configuration file. The default path is /home/admin/.ossutilconfig. If the configuration file (for example, a) exists, ossutil stores a in a.bak, creates file a again, and writes file a to the configuration. If a.bak already exists, it will be overwritten by file a.

NOTE:
(1) If the specified path of the configuration file is not the default path, set the --config-file option to your specified path of the configuration file. (If the --config-file option is not specified, the /home/admin/.ossutilconfig path will be read by default when the command is run.)

(2) Some configuration items can be set using options, such as the --endpoint and --access-key-id options, when a command is run (for details about the options, refer to the help for each command). If you specify the options when running a command and configure the information in the configuration file, the priority is options > configuration file.

(3) If you specify the --endpoint, --access-key-id, --access-key-secret, and --sts-token options when running a command, ossutil does not forcibly require a configuration file.
```

Usage:

This command can be used in 1) interactive mode or 2) non-interactive mode. The interactive mode is recommended because it ensures higher security.

1) ossutil config [-c file]
This mode supports interactive information configuration. Ossutil interactively asks you about the following information:
(1) config file
Specifies the path of a configuration file. If you press Enter, ossutil uses the default configuration file in /home/admin/.ossutilconfig.
If you specify a configuration file, set the --config-file option to the path of your configuration file when running the command. For details about commands that support the --config-file option, refer to the help of each command.
(2) language
During first configuration (the configuration file does not exist), ossutil requires you to set the language. The value can be CH (Chinese) or EN (English). If you press Enter, ossutil configures the language based on the value of the --language option. If you do not set the --language option, ossutil sets the language to CH by default.
If a configuration file exists, ossutil configures the language based on the specified language option and language information in the configuration file.
Ossutil reads the language option from the configuration file during operating. If this option does not exist or is invalid, the ossutil sets the language to CH by default.
NOTE: This configuration item takes effect after the config command is successfully run. When the config command is executed, the displayed language is not affected by your configuration.
(3) endpoint, accessKeyID, accessKeySecret
Enter indicates that a configuration item is skipped. NOTE: The endpoint should be a second-level domain (SLD), for example, oss.aliyuncs.com.
The preceding options are mandatory.
(4) stsToken
To access the OSS using a temporary token, specify this option. Otherwise, press Enter to skip this option.
(5) outputDir
This option is used to configure the path of the directory where the output files reside. In interactive mode, configuration of this option is not supported. However, this option is valid in the configuration file.
The default directory of the outputDir option is ossutil_output of the current directory. Ossutil generates all output files in this folder during operating. Currently, the output files include the report files that record operation errors of each file when exceptions occur for batch operations by running the cp command.
For details about the outputDir option and report files, refer to the cp command help.
NOTE: If the outputDir option does not exist, ossutil automatically creates the directory when generating output files. If the outputDir option exists but is not a directory, an error will be reported.

The following interactive Bucket-Endpoint and Bucket-Cname options are removed, but they are still valid in the configuration file.
(6) Bucket-Endpoint
The Bucket-Endpoint option is used to independently configure the endpoint for each specified bucket. This option is prior to the default endpoint configuration in the configuration file.
In this version, ossutil removes the Bucket-Endpoint pair configuration in interactive mode. However, this configuration item is still valid in the configuration file. Therefore, if you want to independently specify the endpoint for each bucket, you can make configuration in the configuration file. NOTE: The endpoint should be an SLD, for example, oss.aliyuncs.com.
If the Bucket-Endpoint option is specified, ossutil searches for the endpoint corresponding to a bucket in the option when performing operations on the bucket. If being found, the endpoint will overwrite the endpoint in the basic configuration. However, if the --endpoint option is specified when the command is run, the --endpoint option has the highest priority.
(7) Bucket-Cname
The Bucket-Cname option is used to independently configure the CNAME domain name (CDN domain) for each

specified bucket. This option is prior to the configurations of the Bucket-Endpoint option and endpoint in the configuration file.
In this version, ossutil removes the Bucket-Cname pair configuration in interactive mode. However, this configuration item is still valid in the configuration file. Therefore, if you want to independently specify the CNAME domain name for each bucket, you can make configuration in the configuration file.
If the Bucket-Cname option is specified, ossutil searches for the CNAME domain name corresponding to a bucket in the option when performing operations on the bucket. If being found, the CNAME domain name will overwrite the endpoints in the Bucket-Endpoint option and basic configuration. However, if the --endpoint option is specified when the command is run, the --endpoint option has the highest priority.

Priority: --endpoint > Bucket-Cname > Bucket-Endpoint > endpoint > default endpoint

2) ossutil config options
If you specify any options except the --language and --config-file options when running the command, the command enters the non-interactive mode. All configuration items are specified using options.

Configuration file format:

```
[Credentials]
language = CH
endpoint = oss.aliyuncs.com
accessKeyID = your_key_id
accessKeySecret = your_key_secret
stsToken = your_sts_token
outputDir = your_output_dir
[Bucket-Endpoint]
bucket1 = endpoint1
bucket2 = endpoint2
...
[Bucket-Cname]
bucket1 = cname1
bucket2 = cname2
...
```

SAMPLE

```
ossutil config
ossutil config -e oss-cn-hangzhou.aliyuncs.com -c ~/.myconfig
```

OPTIONS

-c, --config-file
Specifies the configuration file path of ossutil. Ossutil reads configuration from the configuration file during startup and writes configuration to the file using the config command.

-e, --endpoint
Specifies the basic endpoint configuration of ossutil (the option value will overwrite the corresponding settings in the configuration file). It must be an SLD.

-i, --access-key-id
Specifies the AccessKeyID used to access the OSS (the option value will overwrite the corresponding settings in the configuration file).

-k, --access-key-secret
Specifies the AccessKeySecret used to access the OSS (the option value will overwrite the corresponding settings in

```
the configuration file).

-t, --sts-token
Specifies the STSToken used to access the OSS (the option value will overwrite the corresponding settings in the configuration file). It is optional.

--output-dir=ossutil_output
Specifies the directory in which output files are located. The output files include the report files generated when errors occur for copying files in batches using the cp command. (For details about the report files, refer to the cp command help.) The default value is the ossutil_output sub-directory in the current directory.

-L CH, --language=CH
Specifies the language of ossutil. The value can be CH or EN, and the default value is CH. If the value is CH, ensure that your system is UTF-8 encoded.
```

Configure ossutil

When using a command to access the OSS, configure the access key pair first. For details about the access key pair, refer to RAM and STS introduction.

Ossutil can be configured to interactive mode or non-interactive mode.

Run ossutil help config to view the help document of the configuration command.

Configure ossutil in interactive mode

./ossutil config

```
$./ossutil config -L ch
This command is used to create a configuration file and store configuration information in it.

You can specify the path for storing the configuration file. The default path is /home/admin/.ossutilconfig. If you press Enter, the default path will be used. If you specify another path, set the --config-file option to this path when running the command.
```

Configure ossutil in non-interactive mode

./ossutil config -e oss.aliyuncs.com -i your_id -k your_key

View all supported options

You can use the -h option to view all options supported by ossutil.

```
$./ossutil -h
Usage of ossutil:
Options:
-s --short-format Used to display the short format. If this option is not specified, the long format is displayed by default.
```

--snapshot-path= Used to accelerate incremental uploading of files in batches in some scenarios. (File downloading and copying do not support this option currently.) This option is used when ossutil uses the cp command to upload files. Ossutil takes a snapshot of file uploads and stores it in the specified directory. It will read the snapshot from the specified directory for incremental upload when this option is used the next time. The specified snapshot directory must be a writable directory in the local file system. If the directory does not exist, ossutil creates a file to record the snapshot information. If the directory already exists, ossutil reads the snapshot information in the directory, performs incremental uploading accordingly, and updates the snapshot information. (Ossutil only uploads files that fail to be uploaded last time and have been locally modified.) NOTE: By using this option, the local lastModifiedTime of files that have been successfully uploaded is recorded and compared with that of files to be uploaded next time to determine whether to skip uploading of same files. When using this option, ensure that the corresponding objects on the OSS are not modified during the two uploading periods. In other scenarios than this one, use the --update option to incrementally upload files in batches. In addition, ossutil does not actively delete the snapshot information under snapshot-path. To avoid too much snapshot information, clear snapshot-path when confirming that the snapshot information is useless.
-j --jobs= Specifies the number of concurrent tasks when multiple files are operated. The value ranges from 1 to 10000, and the default value is 5.
-v --version Used to display ossutil version (1.0.0.Beta2) and exit.
--output-dir= Specifies the directory in which output files are located. The output files include the report file generated when an error occurs for copying files in batches using the cp command. (For details about the report file, refer to the cp command help.) The default value is the ossutil_output sub-directory in the current directory.
--parallel= Specifies the number of concurrent tasks operated in a single file. The value ranges from 1 to 10000. The default value is determined by ossutil based on the operation type and file size.
-L --language= Specifies the language of ossutil. The value can be CH or EN, and the default value is CH.
-t --sts-token= Specifies the STSToken used to access the OSS (the option value will overwrite the corresponding settings in the configuration file). It is optional.
-m --multipart Indicates that the operation objects are the incomplete Multipart events in the bucket, rather than the default objects.
-b --bucket Used to operate a bucket, confirming that an operation is for the bucket.
--delete Used to delete an operation.
-e --endpoint= Specifies the basic endpoint configuration of ossutil (the option value will overwrite the corresponding settings in the configuration file). It must be a second-level domain.
-k --access-key-secret= Specifies the AccessKeySecret used to access the OSS (the option value will overwrite the corresponding settings in the configuration file).
--bigfile-threshold= Specifies the threshold for enabling the resumable data transfer for large files. The value ranges from 0 B to 9223372036854775807 B, and the default value is 100 MB.
--retry-times= Specifies the number of retries when an error occurs. The value ranges from 1 to 500, and the default value is 3.
-a --all-type Indicates that the operation objects are the objects and incomplete Multipart events in the bucket.
-r --recursive Indicates a recursive operation. If this option is specified, commands supporting this option will operate on all objects meeting the criteria in the bucket. Otherwise, the commands operate only on a single object specified in the URL.
-f --force Indicates a forcible operation without asking.
-u --update Indicates an update operation.
-c --config-file= Specifies the configuration file path of ossutil. Ossutil reads configuration from the configuration file during startup and writes configuration to the file using the config command.
-i --access-key-id= Specifies the AccessKeyID used to access the OSS (the option value will overwrite the corresponding settings in the configuration file).
--acl= Used to configure the ACL.
-d --directory Used to return files and sub-directories in the current directory, rather than recursively displaying all objects in all sub-directories.
--checkpoint-dir= Specifies the checkpoint directory path (the default value is .ossutil_checkpoint). If a resumable data transfer fails, ossutil automatically creates this directory and records the checkpoint information in the directory. If a resumable data transfer succeeds, ossutil deletes this directory. If this option is specified, ensure that the specified directory can be deleted.
--type= Specifies the calculation type. The value can be crc64 or md5, and the default value is crc64.
-h --help Show usage message

All commands of ossutil supports part of the preceding options. Use the ossutil help command to check options supported by each command.

Bucket-related commands

Ossutil allows you to create, delete, and list buckets, and set the ACL for a bucket. Other management functions related to the bucket are not supported currently. If you need to use these functions, refer to osscmd.

Before running these commands, run the config command to configure the access key pair.

Create a bucket

ossutil mb oss://bucket [--acl=acl] [--storage-class sc] [-c file]

If the ACL is not specified, the bucket has the private permission by default. After a bucket is created, ossutil prints the consumed time and exits. Otherwise, ossutil outputs error information. You can use the --storage-class option to specify the storage mode.

Run ossutil help mb to view help information about creating a bucket.

```
$./ossutil mb oss://test
0.220478(s) elapsed
```

Delete a bucket

Run ossutil help rm to view help information about deleting a bucket.

Notice:

- The **-b** option must be specified for deleting a bucket.
- The deleted bucket may have been re-created by another user and does not belong to you anymore.
- Once deleted, data in the bucket cannot be recovered.

If you bucket does not contain any data:

ossutil rm oss://bucket -b

```
 $./ossutil rm oss://test -b
Do you really mean to remove the Bucket: test(y or N)? y
0.220478(s) elapsed
```

If your bucket contains the object, multipart, or other data, delete all data before deleting the bucket. You can run the following command to delete all data and your bucket:

ossutil rm oss://bucket -bar

Run ossutil help rm to view help information about deleting a bucket.

List buckets

./ossutil ls or ./ossutil ls oss://

You can use the -s option to display the short format. Run ossutil help ls to view more help information.

```
$./ossutil ls
CreationTime Region StorageClass BucketName
2016-10-21 16:18:37 +0800 CST oss-cn-hangzhou Archive oss://go-sdk-test-bucket-xyz-for-object
2016-12-01 15:06:21 +0800 CST oss-cn-hangzhou Standard oss://ossutil-test
2016-07-18 17:54:49 +0800 CST oss-cn-hangzhou Standard oss://ossutilconfig
2016-07-20 10:36:24 +0800 CST oss-cn-hangzhou IA oss://ossutilupdate
2016-11-14 13:08:36 +0800 CST oss-cn-hangzhou IA oss://yyyyy
2016-08-25 09:06:10 +0800 CST oss-cn-hangzhou Archive oss://ztzt
2016-11-21 21:18:39 +0800 CST oss-cn-hangzhou Archive oss://ztztzt
Bucket Number is: 7
0.252174(s) elapsed
```

List files in a bucket

Ossutil can list objects and UploadIDs in a bucket. The objects are displayed by default. You can use the -m option to display UploadIDs and use the -a option to display the objects and UploadIDs simultaneously.

List objects

./ossutil ls oss://bucket

```
$./ossutil ls oss://ossutil-test
LastModifiedTime Size(B) StorageClass ETAG ObjectName
2016-12-01 15:06:37 +0800 CST 10363812 Standard 61DE142E5AFF9A6748707D4A77BFBCFB oss://ossutil-test/a1
2016-12-01 15:06:42 +0800 CST 10363812 Standard 61DE142E5AFF9A6748707D4A77BFBCFB oss://ossutil-test/a2
2016-12-01 15:06:45 +0800 CST 10363812 Standard 61DE142E5AFF9A6748707D4A77BFBCFB oss://ossutil-test/a3
Object Number is: 3
0.007379(s) elapsed
```

List objects and multiparts

./ossutil ls oss://bucket -a

```
$ ossutil ls oss://bucket1 -a
LastModifiedTime Size(B) StorageClass ETAG ObjectName
2015-06-05 14:06:29 +0000 CST 201933 Standard 7E2F4A7F1AC9D2F0996E8332D5EA5B41
oss://bucket1/dir1/obj11
2015-06-05 14:36:21 +0000 CST 201933 Standard 6185CA2E8EB8510A61B3A845EAFE4174 oss://bucket1/obj1
2016-04-08 14:50:47 +0000 CST 6476984 Standard 4F16FDAE7AC404CEC8B727FCC67779D6
oss://bucket1/sample.txt
Object Number is: 3
InitiatedTime UploadID ObjectName
2017-01-13 03:45:26 +0000 CST 15754AF7980C4DFB8193F190837520BB oss://bucket1/obj1
2017-01-13 03:43:13 +0000 CST 2A1F9B4A95E341BD9285CC42BB950EE0 oss://bucket1/obj1
2017-01-13 03:45:25 +0000 CST 3998971ACAF94AD9AC48EAC1988BE863 oss://bucket1/obj2
2017-01-20 11:16:21 +0800 CST A20157A7B2FEC4670626DAE0F4C0073C oss://bucket1/tobj
UploadId Number is: 4
0.191289(s) elapsed
```

You can use the -s option to display the short format.

You can use the -d option to display content in the level 1 directory.

```
$ ossutil ls oss://bucket1 -d
oss://bucket1/obj1
oss://bucket1/sample.txt
oss://bucket1/dir1/
Object and Directory Number is: 3
UploadID ObjectName
15754AF7980C4DFB8193F190837520BB oss://bucket1/obj1
2A1F9B4A95E341BD9285CC42BB950EE0 oss://bucket1/obj1
3998971ACAF94AD9AC48EAC1988BE863 oss://bucket1/obj2
A20157A7B2FEC4670626DAE0F4C0073C oss://bucket1/tobj
UploadId Number is: 4
0.119884(s) elapsed
```

Set the ACL for a bucket

When a bucket is created, the default ACL for the bucket is private. You can run the set-acl command to modify the ACL for a bucket. You need to specify the -b option when setting the ACL for a bucket.

Grant the private permission for bucket1:

./ossutil set-acl oss://bucket1 private -b

Run the help set set-acl command to view more information about setting the ACL.

Object-related commands

Ossutil allows you to upload/download/copy a file, set the ACL and meta of an object, as well as view

the meta information of an object.

Run the config command to configure the access key pair before running these commands.

Upload/Download/Copy a file

You are strongly advised to use ossutil help cp to view the help information before running the cp command.

You can run the cp command to upload/download/copy a file, and use the -r option to copy a folder. Ossutil implements multipart upload by default for large files and supports the resumable data transfer (the threshold of large files for which multipart upload is enabled can be set using the --bigfile-threshold option.)

Use the -f option to forcibly upload a file by default. If a file exists with the same name on the target end, the file will be overwritten directly.

If an error occurs to a file during file uploading/downloading/copying in batches, ossutil logs the error information in the report file by default, skips this file, and performs operations on other files. (Ossutil does not continue to copy other files if the bucket does not exist, or permission verification is invalid due to incorrect accessKeyID/accessKeySecret.) For details, refer to ossutil help cp.

Ossutil supports the incremental uploading policies --update and --snapshot-path in specific scenarios. For details, refer to ossutil help cp.

From ossutil 1.0.0.Beta1, crc64 is enabled by default during file uploading.

(1) Upload a single file:

```
$./ossutil cp a oss://ossutil-test
Succeed: Total num: 1, size: 230. OK num: 1(upload 1 files).
0.699795(s) elapsed
```

(2) Upload a folder:

```
$./ossutil cp -r dir oss://ossutil-test
Succeed: Total num: 35, size: 464,606. OK num: 35(upload 34 files, 1 directories).
0.896320(s) elapsed
```

Performance tuning for uploading/downloading/copying a file

In the cp command, the jobs and -parallel options are used to control the number of concurrent operations. The -jobs option controls the number of concurrent operations enabled between files when multiple files are uploaded/downloaded/copied. The -parallel option controls the number of concurrent operations enabled for a large file when the large file is uploaded/downloaded/copied in multiparts.

Ossutil calculates the number of parallel operations based on the file size by default (this option does

not work for small files, and the threshold for large files to be uploaded/downloaded/copied in multiparts can be controlled by the —bigfile-threshold option). When large files are uploaded/downloaded/copied in batches, the actual number of concurrent operations is calculated by multiplying the number of jobs by the number of parallel operations. If the default number of concurrent operations set by ossutil cannot meet your performance requirements, you can adjust these two options to improve or reduce the performance.

> **Note:**
> If the number of concurrent operations is too large, ossutil’ s uploading/downloading/copying performance may be reduced due to inter-thread resource switching and snatching. Therefore, adjust the values of these two options based on the actual machine conditions. To perform pressure testing, set the two options to small values first, and slowly adjust them to the optimal values.
> If the values of the —jobs and —parallel options are too large, an EOF error may occur due to the slow network transfer speed if machine resources are limited. In this case, appropriately reduce the values of the —jobs and —parallel options.

Configure the ACL of an object

Ossutil uses the set-acl command to configure the ACL of an object. You can use the -r option to configure the ACLs of objects in batches.

For details, refer to ossutil help set-acl.

```
$./ossutil set-acl oss://dest/a private
0.074507(s) elapsed
```

Configure the ACLs of objects in batches:

```
$./ossutil set-acl oss://dest/a private -r
Do you really mean to recursivlly set acl on objects of oss://dest/a(y or N)? y
Succeed: Total 3 objects. Setted acl on 3 objects.
0.963934(s) elapsed
```

Configure the meta of an object

Ossutil uses the set-meta command to configure the meta information of an object. You can use the -r option to configure the metas of objects in batches.

For details, refer to ossutil help set-meta.

```
./ossutil set-meta oss://dest/a x-oss-object-acl:private -u
```

View the object description (meta)

Ossutil uses the stat command to view the object description (meta).

For details, refer to ossutil help stat.

```
$./ossutil stat oss://dest/a
ACL : default
Accept-Ranges : bytes
Content-Length : 230
Content-Md5 : +5vbQC/MSQK0xXSiyKBZog==
Content-Type : application/octet-stream
Etag : FB9BDB402FCC4902B4C574A2C8A059A2
Last-Modified : 2017-01-13 15:14:22 +0800 CST
Owner : aliyun
X-Oss-Hash-Crc64ecma : 12488808046134286088
X-Oss-Object-Type : Normal
0.125417(s) elapsed
```

Restore an object from the frozen state to the readable state

Ossutil uses the restore command to restore an object from the frozen state to the readable state. You can use the -r option to restore objects from the frozen state to the readable state in batches.

For details, refer to ossutil help restore.

```
$./ossutil restore oss://utiltest/a
0.037729(s) elapsed
```

Create a symbolic link

Ossutil uses the create-symlink command to create a symbolic link.

For details, refer to ossutil help create-symlink.

```
$./ossutil create-symlink oss://utiltest/b a
0.037729(s) elapsed
```

Read the description of a symbolic link file

Ossutil uses the read-symlink command to read the description of a symbolic link file.

For details, refer to ossutil help read-symlink.

```
$./ossutil read-symlink oss://utiltest/b
Etag : D7257B62AA6A26D66686391037B7D61A
Last-Modified : 2017-04-26 15:34:27 +0800 CST
```

```
X-Oss-Symlink-Target : a
0.112494(s) elapsed
```

Multipart-related commands

Ossutil allows you to list an UploadID and delete all UploadIDs of the specified object. Other management functions related to the Multipart UploadID are not supported currently. If you need to use these functions, refer to osscmd.

For details about the multipart, refer to Multipart upload.

> Note: When uploading/copying a large file, ossutil automatically implements multipart upload and resumable data transfer, without running the UploadPart command.

List an UploadID

Use the -m option to list all incomplete UploadIDs of the specified object, and use the -a option to list objects and UploadIDs.

```
$ ossutil ls oss://bucket1/obj1 -m
InitiatedTime UploadID ObjectName
2017-01-13 03:45:26 +0000 CST 15754AF7980C4DFB8193F190837520BB oss://bucket1/obj1
2017-01-13 03:43:13 +0000 CST 2A1F9B4A95E341BD9285CC42BB950EE0 oss://bucket1/obj1
UploadId Number is: 2
0.070070(s) elapsed
```

Delete all UploadIDs of the specified object

Use the -m option to delete all incomplete UploadIDs of the specified object. If the -r option is specified simultaneously, incomplete UploadIDs of all objects that use the specified object as the prefix are deleted.

Assume that bucket1 contains the following objects:

```
$ ossutil ls oss://bucket1 -a
LastModifiedTime Size(B) StorageClass ETAG ObjectName
2015-06-05 14:06:29 +0000 CST 201933 Standard 7E2F4A7F1AC9D2F0996E8332D5EA5B41
oss://bucket1/dir1/obj11
2015-06-05 14:36:21 +0000 CST 241561 Standard 6185CA2E8EB8510A61B3A845EAFE4174 oss://bucket1/obj1
2016-04-08 14:50:47 +0000 CST 6476984 Standard 4F16FDAE7AC404CEC8B727FCC67779D6
oss://bucket1/sample.txt
Object Number is: 3
InitiatedTime UploadID ObjectName
2017-01-13 03:45:26 +0000 CST 15754AF7980C4DFB8193F190837520BB oss://bucket1/obj1
```

```
2017-01-13 03:43:13 +0000 CST 2A1F9B4A95E341BD9285CC42BB950EE0 oss://bucket1/obj1
2017-01-13 03:45:25 +0000 CST 3998971ACAF94AD9AC48EAC1988BE863 oss://bucket1/obj2
2017-01-20 11:16:21 +0800 CST A20157A7B2FEC4670626DAE0F4C0073C oss://bucket1/tobj
UploadId Number is: 4
0.191289(s) elapsed
```

Delete the two UploadIDs of obj1:

```
$./ossutil rm -m oss://bucket1/obj1
Succeed: Total 2 uploadIds. Removed 2 uploadIds.
1.922915(s) elapsed
```

Delete the three UploadIDs of obj1 and obj2:

```
$./ossutil rm -m oss://bucket1/ob
Succeed: Total 4 uploadIds. Removed 4 uploadIds.
1.922915(s) elapsed
```

Delete obj1 and the three UploadIDs of obj1 and obj2 simultaneously:

```
$./ossutil rm  oss://dest1/.a  -a -r -f
Do you really mean to remove recursively objects and multipart uploadIds of oss://dest1/.a(y or N)? y
Succeed: Total 1 objects, 3 uploadIds. Removed 1 objects, 3 uploadIds.
```

ossftp

Quick installation

Introduction

The OSS FTP is a special FTP server that maps the operations on files and folders into your OSS instance upon receiving a common FTP request. This utility allows you to use the FTP protocol to manage files stored on your OSS instance.

Key features

- **Cross-Platform:** This utility can run on Windows, Linux, and Mac operating systems, either 32 or 64 bit, either on a graphic or command-line interface.

- **Free of Installation:** You can run this utility directly after extraction.
- **Free of Configuration:** You can run the utility without any further configurations.
- **Transparent:** The FTP utility was written in Python, so you can see the complete source code. We will soon make the open source available on GitHub.

Key functions

- Supports file/folder upload, download, delete, and other operations
- Supports multipart upload of large files
- Supports most FTP commands and can satisfy daily needs

Note

1. Currently, for the ease of installation and deployment, OSS FTP V1.0 does not support TLS encryption. The FTP protocol implements plaintext transmission. **To prevent password leaks, we recommend that you run the FTP server and client on the same machine** and access using 127.0.0.1:port.
2. The utility does not support rename and move operations.
3. Do not include any Chinese characters in the extract-to path of the installation package.
4. The FTP server’ s management control page may fail to be opened on early IE browsers.
5. Supported Python versions: Python 2.6 and Python 2.7

Downloads

- Windows: ossftp-1.0.2-win.zip

Now that Python 2.7 is not installed on Windows by default, it is contained in the installation package and is ready for use after extraction, without the hassle of installation and configuration.

- Linux/Mac: ossftp-1.0.2-linux-mac.zip

Because Python 2.7 or 2.6 is installed on Linux and Mac systems by default, the installation packages for Linux and Mac do not contain an executable Python program, but only relevant dependent libraries.

Running

First, extract the downloaded file. Then, select an appropriate running mode based on environmental conditions.

- Windows: Double-click start.vbs to run it.

- Linux: Start the terminal and run it.

```
$ bash start.sh
```

- Mac: Double-click start.command or run it on a terminal.

```
$ bash start.command
```

The preceding process starts an FTP server, which listens to port 2048 at 127.0.0.1 by default.
In addition, for ease of control over the status of the FTP server, the program also activates a web server, which listens to port 8192 at 127.0.0.1.
If your system has a graphic interface, the control page will be automatically opened.

> In most situations, you do not need to configure any settings before running the FTP server. If you make any configuration, remember to restart it to make the changes take effect.

Connecting to the FTP Server

We recommend using the FileZilla Client to connect to the FTP server. After download and installation, connect to the FTP server as follows:

- Host: 127.0.0.1
- Login type: normal
- User: access_key_id/bucket_name (The slash sign (/) means that both, not either items are required. For example, the user could be ‘tSxyiUM3NKswPMEp/test-hz-jh-002’.)
- Password: access_key_secret

Advanced use

Manage the ftpserver from the console page

Modify the Listener Address
If you need to access the ftpserver over a network, you must modify the listener address because the default address, 127.0.0.1, only allows local access. You can change it to an intranet IP or Internet IP.

Modify the Listening Port
Modify the ftpserver’s listening port. We suggest using a port over 1024 because ports below 1024 require administrator permissions.

Modify the Log Level
Set the ftpserver’ s log level. The ftpserver’ s log is output to the data/ossftp/ directory. You can view it only by pressing the Log button on the console page. The default log level is INFO and little information is printed in the log. If you need more detailed log information, you can change the level to DEBUG. If you want the log to output less information, you can set the level to WARNING or ERROR.

Set Bucket Endpoints
By default, the ftpserver will search for the bucket’ s location information, so it can send subsequent requests to the corresponding region (such as oss-cn-hangzhou.aliyuncs.com or oss-cn-beijing.aliyuncs.com). The ftpserver will first try to access the OSS instance over the intranet.
If you set bucket endpoints, for example, ‘test-bucket-a.oss-cn-hangzhou.aliyuncs.com’ , when you access test-bucket-a, you will go to the ‘oss-cn-hangzhou.aliyuncs.com’ domain name.

Note : The system must be restarted for modifications to take effect.
All the above modifications are actually changes to the ftp directory’ s config.json file. Thus, you can also modify this file directly.

Directly launch ftpserver (Linux/Mac)You can simply launch the ftpserver.py file in the ossftp directory to avoid web_server overhead.

```
$ python ossftp/ftpserver.py &
```

The configuration modification method is the same as above.

Possible problems

If you encounter an error when connecting to the FTP server.

There are two possible causes:

There may be an error in the entered access_key_id or access_key_secret.Solution: Enter the correct information and try again.

The used access_key information may be a RAM sub-account access_key for a sub-account without list buckets permission.

Solution: When using a sub-account, specify bucket endpoints on the console page

to tell the ftpserver which endpoint should be used to access a certain bucket. Also, the sub-account must have the required permissions. For information on implementing access control by using RAM to access OSS, refer to RAM. The details about permissions are as follow:

> **Read-only**: The OSS-FTP must have these permissions:['ListObjects' , 'GetObject' , 'HeadObject'].For information on creating a RAM sub-account with **Read-only** permission, refer to the graphic tutorial How to Integrate RAM for File Sharing.
>
> If you want to allow a RAM sub-account to **upload files**, assign ['PutObject'] permission.
>
> If you want to allow a RAM sub-account to **delete files**, assign ['DeleteObject'] permission.

If you are running the FTP server on Linux, you may encounter the following error when using FileZilla to connect to the server:

501 can't decode path (server filesystem encoding is ANSI_X3.4-1968)

This is usually generated when errors occur in local Chinese code. Input the following command in the terminal where you want to run start.sh. Then, restart the program.

```
$ export LC_ALL=en_US.UTF-8; export LANG="en_US.UTF-8"; locale
```

How to store remote attachments to your OSS instance with Discuz

Preface

The website remote attachment function refers to directly storing uploaded attachments to a remote storage server, which is usually a remote FTP server, over the FTP.

Currently, Discuz forums, PHPWind forums, and WordPress websites support the remote attachment function.

This document instructs you on storing remote attachments from a Discuz-based forum.

Preparation

Apply for an OSS account and create a **public-read** bucket. You must set the permission to public-read because it must allow anonymous access.

Procedures

Here the Discuz version we use is **Discuz! X3.1** and the detailed configuration process is shown below.

Log on to the Discuz website and go to the management interface. Click **Global** and then **Upload Settings**.

Select **Remote Attachments** and configure the function.

Set "Enable remote attachment" to **Yes**.

Set "Enable SSL connection" to "**No** ".

Set the "FTP Server Address", that is, the address that runs the OSS-FTP. Generally, this is "**127.0.0.1** ".

Set "FTP service port No." to the default "**2048** ".

Set "FTP Account" in the format of **AccessKeyID/BucketName**, where "**/** " does not mean "**or** ".

Set "FTP Password" to **AccessKeySecret**.

Set "Passive Mode Connection" to the default "**Yes** ".

Set "Remote Attachment Directory" to "**.** ", that is, to create a directory for upload under the root directory of the bucket.

Set "Remote URL" to http://BucketName.Endpoint.

Here, we will test the bucket test-hz-jh-002 from the Hangzhou region. Therefore, we enter http://test-hz-jh-002.oss-cn-hangzhou.aliyuncs.com, **where the BucketName must match the endpoint**.

Set the timeout time to 0, that is, to use the default setting of the service.

After the configuration is complete, click “Test Remote Attachment”. If the test is successful, an information box will be displayed.

- Verification

Ok, now let’s publish a post on the forum to test the function. On any board, create a post and upload an image as attachment in the post.

Right-click the image and select “Open image in new tab”.

In the browser, you can see the image URL is http://test-hz-jh-002.oss-cn-hangzhou.aliyuncs.com/forum/201512/18/171012mzvkku2z3na2w2wa.png. This indicates that the image has been uploaded to test-hz-jh-002 in the OSS.

How to store remote attachments to your OSS instance with PHPWind

Preface

The website remote attachment function refers to directly storing uploaded attachments to a remote storage server, which is usually a remote FTP server, over the FTP.

Currently, Discuz forums, PHPWind forums, and WordPress websites support the remote attachment function.

This document instructs you on storing remote attachments from a PHPWind-based forum.

Preparation

Apply for an OSS account and create a **public-read** bucket. You must set the permission to public-read because it must allow anonymous access.

Procedures

The PHPWind we use is **PHPWind 8.7** and the configuration process is as follows.

Log on to the website.

Go to the management interface and select **Global** -> **Upload Settings** -> **Remote Attachments**.

Configure the function.

i. Set "**Enable FTP uploads**" to "**Yes**".
ii. Set "**Website Attachment Address**" to "http://bucket-name.endpoint". Here, we will test the bucket test-hz-jh-002 from the Hangzhou region. Therefore, we enter http://test-hz-jh-002.oss-cn-hangzhou.aliyuncs.com, **where the BucketName must match the endpoint**.
iii. Set the FTP server address, that is, the address that runs the OSS-FTP. Generally, this is **127.0.0.1**.
iv. Set "FTP service port No." to the default "**2048**".
v. Set "Remote attachment directory" to "**.**", that is, to create a directory for upload under the root directory of the bucket.
vi. Set "FTP Account" in the format of **AccessKeyID/BucketName**, where "**/**" does not mean "**or**".
vii. Set "FTP Password" to **AccessKeySecret**. To obtain the AccessKeyID and AccessKeySecret, you can log on to the Alibaba Cloud console and go to Access Key Management.
viii. Set the FTP timeout time. If you set it to "10", a timeout response is sent if a request does not receive a response within 10 seconds.

Verification

PHPWind does not allow users to directly test the function by clicking a test button. Therefore, we must publish a post with an image to verify the function.

Right-click the image and select "Open image in new tab". The image is displayed in a new tab.

The image URL indicates that the image has been uploaded to bucket test-hz-jh-002 in the OSS.

How to store remote attachments to your OSS instance with WordPress

Preface

The website remote attachment function refers to directly storing uploaded attachments to a remote storage server, which is usually a remote FTP server, over the FTP.

Currently, Discuz forums, PHPWind forums, and WordPress websites support the remote attachment function.

This document instructs you on storing remote attachments from a WordPress-based forum.

Preparation

Apply for an OSS account and create a **public-read** bucket. You must set the permission to public-read because it must allow anonymous access.

Procedures

WordPress does not have inherent support for this function, but implements remote attachment using a third-party plug-in. The WordPress we use is WordPress **4.3.1** and the plug-in is **Hacklog Remote Attachment**. The specific configuration process is as follows:

1. Log on to the WordPress website and select “Install Plug-in”. Search for the keyword “FTP” and choose to install **Hacklog Remote Attachment**.

   Configuration.

   i. Set the FTP server address, that is, the address that runs the OSS-FTP. Generally, this is **127.0.0.1**.
   ii. Set “FTP service port No.” to the default “**2048**”.
   iii. Set “FTP Account” in the format of **AccessKeyID/BucketName**, where “**/**” does not mean “**or**”.
   iv. Set “FTP Password” to **AccessKeySecret**.

   > To obtain the AccessKeyID and AccessKeySecret, you can log on to the Alibaba Cloud console and go to Access Key Management.

   v. Set the FTP timeout to the default value, 30 seconds.
   vi. Set “Remote Basic URL” to http://BucketName.Endpoint/wp. Here, we will test the bucket test-hz-jh-002 from the Hangzhou region. Therefore, we enterhttp://test-hz-jh-002.oss-cn-hangzhou.aliyuncs.com/wp
   vii. Set “FTP Remote Path”. We enter “wp”, that is, to save all attachments to the bucket’s wp directory.Note that this field is related to the “Remote Basic URL” field.
   viii. Set “HTTP Remote Path” to “.”.

Verification.
After the configuration is complete, click “Save” and a test starts automatically. The test results are shown at the top of the page.

Post a new article and insert an image.
Now you can write a new article and test the remote attachment function. After creating an article, click “Add Media” to upload an attachment.

When the attachment is uploaded, click “Post” to view your article.

Right-click the image and click “Open image in new tab” to see the image URL.

The image URL indicates that the image has been successfully uploaded to the OSS.

How to integrate RAM for file sharing

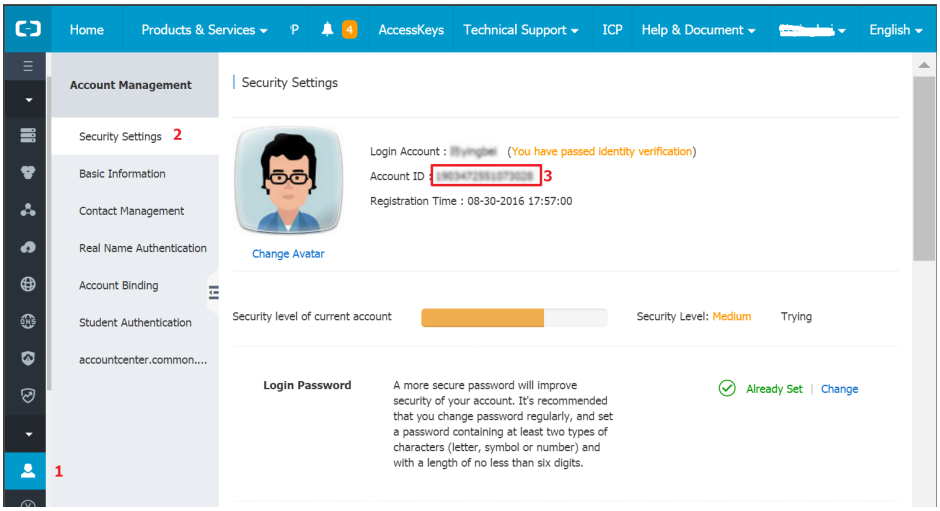
Introduction

This document instructs you on integrating the RAM service to share files and folders in user buckets. Other users will have read-only permission, while the bucket owner can edit the objects.

> Process: Activate RAM -> Create a read-only authorization policy -> Create sub-accounts -> Grant permissions to the sub-accounts -> Verify FTP logon

Retrieve account ID

Retrieve your account ID, as shown in the image below:

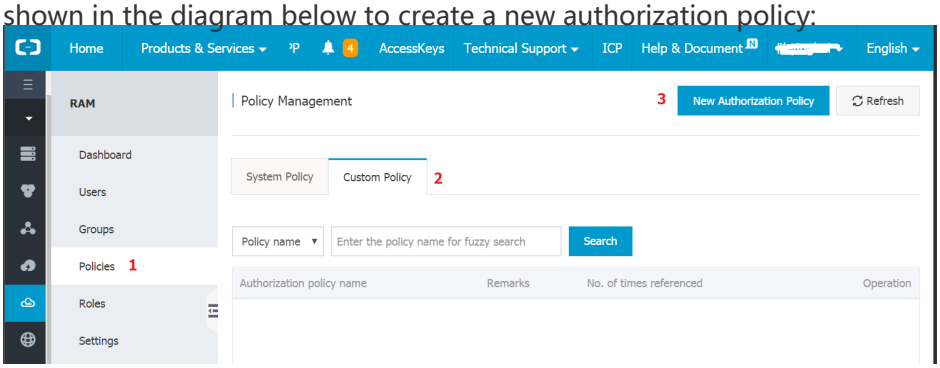


Activate RAM

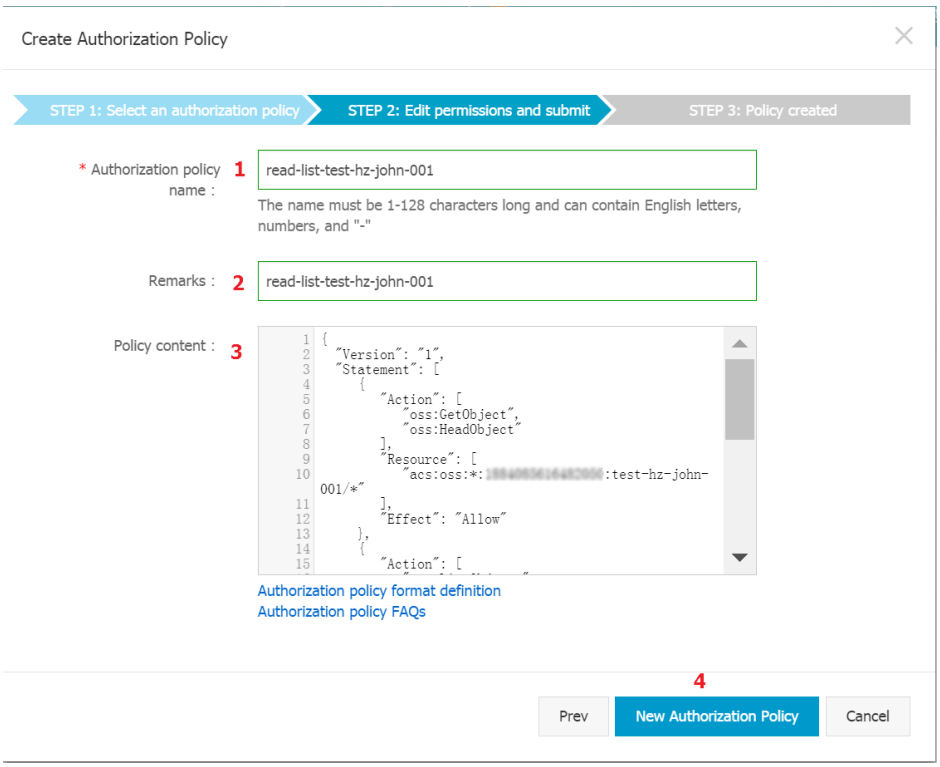
Resource Access Management (RAM) is an Alibaba Cloud service designed for controlling resource access. By creating a policy, you can create a shared read account. Users can use this account to log on to the FTP tool and read your files.

Create an authorization policy

After activating RAM, go to the RAM console and click “Policies” on the left side. Follow the steps shown in the diagram below to create a new authorization policy:



Enter the authorization policy as shown below:



Specify policy name and remarks (fields 1 and 2) as needed. “Policy content” in field 3 determines the policy.

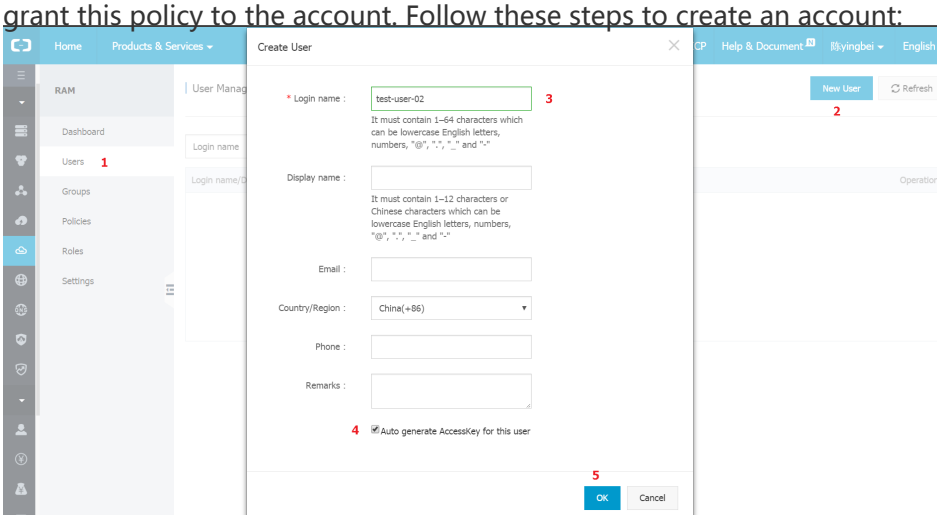
```
{
"Version": "1",
"Statement": [
{
"Action": [
"oss:GetObject",
"oss:HeadObject"
],
"Resource": [
"acs:oss:*:****************:test-hz-john-001/*"
],
"Effect": "Allow"
},
{
"Action": [
"oss:ListObjects",
"oss:GetBucketAcl",
"oss:GetBucketLocation"
],
"Resource": [
"acs:oss:*:****************:test-hz-john-001"
],
"Effect": "Allow"
},
{
"Action": [
"oss:ListBuckets"
],
```

```
"Resource": [
"acs:oss:*:**********:*"
],
"Effect": "Allow"
}
]
}
```

In the example above, replace ********** with your own account ID and replace test-hz-john-001 with your bucket name. Then, copy all the content and paste it in “Policy content”. Finally, click “New Authorization Policy”.

Create an account

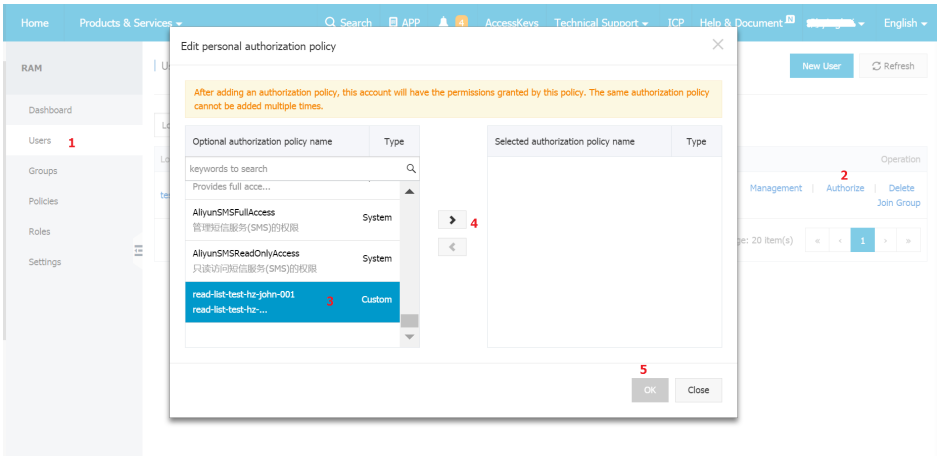
The above authorization policy produces a read-only policy. Below, we will create an account and grant this policy to the account. Follow these steps to create an account:



Remember to record the new account’s access_key.

Authorize the account

Below, we will grant the new policy to the account.



Log on with the sub-account

Use the sub-account’ s access_key and the bucket in the authorization policy to log on. Now, you can download files and folders, but upload operations will fail.

ossfs

Quick installation

Introduction

ossfs allows you to mount Alibaba Cloud OSS buckets to local files in Linux systems. In the system, you can quickly use the local file system to perform operations on OSS objects, achieving data sharing.

Functions

The ossfs is constructed based on S3FS and incorporates all S3FS functions. Key functions include:

- Support for most functions of the POSIX file system, including file reading/writing, directories, link operations, permissions, UID/GID, and extended attributes.
- Uploads of large files using the OSS multipart function.
- MD5 checksum to ensure data integrity.

Limitations

Compared to a local file system, the functions and performance provided by ossfs have certain limitations. These include:

- Random write and append operations will overwrite the entire file.
- The performance of metadata operations, such as list directory, is poor because the system has to remotely access the OSS server.
- The file/folder rename operation is not atomic.
- When multiple clients are attached to a single OSS bucket, you must coordinate the actions of each client manually. For example, you need to avoid multiple clients writing the same file.
- Hard link is not supported.
- This system is not suitable for highly-concurrent read/write scenarios, as this will greatly increase the system load.

Installation and use

Installation package download:

Released Linux	Download
Ubuntu 16.04 (x64)	ossfs_1.80.2_ubuntu16.04_amd64.deb
Ubuntu 14.04 (x64)	ossfs_1.80.2_ubuntu14.04_amd64.deb
CentOS 7.0 (x64)	ossfs_1.80.2_centos7.0_x86_64.rpm
CentOS 6.5 (x64)	ossfs_1.80.2_centos6.5_x86_64.rpm

Install the ossfs

- Run these commands to install Ubuntu:

```
sudo apt-get update
sudo apt-get install gdebi-core
sudo gdebi your_ossfs_package
```

- Run these commands to install CentOS 6.5 or above:

```
sudo yum localinstall your_ossfs_package
```

- Run these commands to install CentOS 5 or above:

```
sudo yum localinstall your_ossfs_package --nogpgcheck
```

Use the ossfs

Set bucket name and AccessKeyId/Secret and save it to the /etc/passwd-ossfs file.

Note that the permissions for this file must be set correctly. We suggest setting it to 640.

```
echo my-bucket:my-access-key-id:my-access-key-secret > /etc/passwd-ossfs
chmod 640 /etc/passwd-ossfs
```

Mount the OSS bucket to the specified directory.

```
ossfs my-bucket my-mount-point -ourl=my-oss-endpoint
```

Example

Mount the bucket my-bucket to the /tmp/ossfs directory. The AccessKeyId is faint,the AccessKeySecret is 123, and the OSS endpoint is http://oss-cn-hangzhou.aliyuncs.com.

```
echo my-bucket:faint:123 > /etc/passwd-ossfs
chmod 640 /etc/passwd-ossfs
mkdir /tmp/ossfs
ossfs my-bucket /tmp/ossfs -ourl=http://oss-cn-hangzhou.aliyuncs.com
```

Unmount the bucket:

```
fusermount -u /tmp/ossfs
```

For more information, refer to https://github.com/aliyun/ossfs#ossfs.

Release log

Refer to https://github.com/aliyun/ossfs/blob/master/ChangeLog.

FAQ

- Q: For what programs is ossfs suitable?
    - ossfs mounts OSS buckets locally. If you want a program that does not support OSS to automatically sync the data to the OSS, ossfs is a great option.
- Q: What are the limitations of ossfs?
    - Because data must be synced to the cloud over the network, the performance and functions of ossfs may differ from those of local file systems. If you want to run a

database or other applications with frequent I/O operations on a mounted ossfs disk, you must consider this carefully. ossfs differs from local file systems in the following ways:

- Random write and append operations will overwrite the entire file.
- The performance of metadata operations, such as list directory, is poor because the system has to remotely access the OSS server.
- The file/folder rename operation is not atomic.
- When multiple clients are attached to a single OSS bucket, you must coordinate the actions of each client manually. For example, you need to avoid multiple clients writing the same file.
- Hard link is not supported.

- Q: Do I need to use Alibaba Cloud hosts for ossfs?
  - ossfs does not need to be used with Alibaba Cloud intranet. It can be used on external Internet hosts.
- Q: Can ossfs simultaneously mount multiple OSS buckets?
  - Yes, just write multiple OSS configuration information entries in the passwd-ossfs file. Buckets from different OSS accounts are supported.
- Q: When trying to mount a bucket, why do I receive the error “ossfs: unable to access MOUNTPOINT /tmp/ossfs: Transport endpoint is not connected”?
  - First, run the umount command for the corresponding directory.
  - When mounting with ossfs, check that the entered URL parameter is correct and the bucket, access key ID, and access key secret match.
  - DO NOT include the bucket name in the URL. For example, if the bucket domain name isossfs-test-1.oss-cn-hangzhou.aliyuncs.com on the OSS console, set the URL to http://oss-cn-hangzhou.aliyuncs.com.
- Q: Why does ossfs display “ossfs: unable to access MOUNTPOINT /tmp/odat: No such file or directory”?
  - This error occurs if the directory is not yet created. You must create the directory before mounting.
- Q: Why does the “operation not permitted” error occur after I mount the bucket locally and run the ls command for the directory?
  - In your bucket, check if the directory name contains any OSS objects with invisible characters. The file system has strict restrictions for file/directory names. If the directory name fails to meet the restrictions, this error occurs. Use another tool to rename these objects and run the ls command, the directory content can be correctly displayed.
- Q: How do I set permissions during ossfs mounting?
  - If you want to allow other users to access mounted folders, specify the allow_other parameter as follows when running ossfs:
    - ossfs your_bucket your_mount_point -ourl=your_endpoint -o allow_other
  - If you want to allow the mounting of folders (/tmp/ossfs) that belong to another user, you must create and mount a folder and use OSSFS as this user:
    - sudo -u user mkdir /tmp/ossfs

- sudo -u user ossfs bucket-name /tmp/ossfs

- Q: How can I mount ossfs automatically when the device starts up?
  - Step 1: Write the bucket name, access key ID/secret, and other information into /etc/passwd-ossfs, and change the permissions for this file to 640.
    - echo your_bucket_name:your_access_key_id:your_access_key_secret > /etc/passwd-ossfs
    - chmod 640 /etc/passwd-ossfs

  Step 2: Make the appropriate settings (the setting methods differ for different system versions).

  - Step 2A: Use the fstab method to automatically mount the ossfs (applies to Ubuntu 14.04 and CentOS 6.5).
    - Add the following command in /etc/fstab:
    - ossfs#your_bucket_name your_mount_point fuse _netdev,url=your_url,allow_other 0 0
    - In the above command, replace ‘your_xxx’ with your actual bucket name and other information.
    - Save the /etc/fstab file. Execute the mount -a command. If no error is reported, the settings are correct.
    - Now, Ubuntu 14.04 can automatically mount the ossfs. For CentOS 6.5, also execute the following command:
    - chkconfig netfs on
  - Step 2B: Mount ossfs using a boot script (applies to CentOS 7.0 and above).
    - Create the file ossfs in the /etc/init.d/ directory. Copy the content in the Template File to the new file. Here, replace ‘your_xxx’ with your own information.
    - Execute the command: chmod a+x /etc/init.d/ossfs.
    - The above command will grant execution permission to the new ossfs script. You can now execute this script. If there are no errors in the script content, the OSS bucket has been mounted to the specified directory.
    - Execute the command: chkconfig ossfs on.
    - The above command sets the ossfs boot script as another service, so it will be automatically started when the device starts up.
    - ossfs can now automatically mount upon startup. To sum up, if you use Ubuntu 14.04 or CentOS 6.5, perform Steps 1 and 2A; if you use CentOS 7.0, perform Steps 1 and 2B.

Q: I need to use a www user to mount ossfs. In this case, how do I set up automatic mounting?

Refer to the answer to the question above. Perform Step 1 as stated. Perform Step 2B with the command in the /etc/init.d/ossfs file changed to:

sudo -u www ossfs your_bucket your_mountpoint -ourl=your_url

- Set the boot script to allow the use of sudo to edit /etc/sudoers. Change the Defaults requiretty line to #Defaults requiretty (comment out this line).

- Q: How do I solve the fusermount: failed to open current directory: Permission denied error?
  - This is a fuse bug. It requires the current user to have read permission for the current directory (unmounted directory). To solve this problem, run the cd command to change to a directory with read permission and then run the ossfs command again.

osscmd

Quick installation

Environment requirement

Python SDK requires a Python-ready environment. Python versions: Version 2.5 to Version 2.7. SDK is applicable to Windows and Linux, but as Python3.0 is not fully compatible with SDK Version 2.x, SDK does not support Python3.0 or above.

After Python is installed:

- Input **python** in Linux shell and press Enter to view the Python version. As shown below:

```
Python 2.5.4 (r254:67916, Mar 10 2010, 22:43:17)
[GCC 4.1.2 20080704 (Red Hat 4.1.2-46)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

- Input **python** in Windows cmd and press Enter to view the Python version. As shown below:

```
C:\Documents and Settings\Administrator>python
Python 2.7.5 (default, May 15 2013, 22:43:36) [MSC v.1500 32 bit (Intel)] on win
32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

The above shows the Python has been installed successfully.

Exception: After entering **python** in Windows cmd and pressing Enter, the system prompts **Not an**

internal or external command. In such a case, check the configuration **Environment variables** > **Path** and add the Python installation path.

If the Python is not installed, you can get its installer from Python official website. The website provides detailed instructions and guidance for installing and using Python.

Installation and usage

Click here to view how to download the file. Unzip the downloaded Python SDK to the directory of the osscmd and then execute python osscmd + operation. For example, upload an object to the bucket:

```
python  osscmd  put  myfile.txt  oss://mybucket
```

Please note that in osscmd, we use oss://bucket or oss://bucket/object to indicate a bucket or an object. oss:// is merely a way to indicate the resource with no other meanings.

If you need the detailed command list, enter: python osscmd.

If you need the detailed parameter list instructions, enter: python osscmd help.

Example

Install and configure osscmd

After you download SDK installer in Linux or Windows, unzip the downloaded packet to start using osscmd.

You can directly invoke python osscmd to get instructions for use. Every command has two modes for execution. Take querying the user-created bucket for example. The gs command (short for “get service”) will be executed.

- Method 1: No ID or Key is specified, and osscmd will read the ID and Key from default files.

```
$ python osscmd gs
can't get accessid/accesskey, setup use : config --id=accessid --key=accesskey
```

> **Notice:** In the case of such prompts, it indicates that the ID and Key are not properly configured. See the configuration command in Step 2.

Once the ID and Key are properly configured and valid, run the command

```
$ python osscmd gs
2013-07-19 08:11 test-oss-sample
Bucket Number is: 1
```

- Method 2: Specify the ID and Key in the command and osscmd will read ID and Key from the command line. If the ID and Key are valid, run the command and the following result will show.

```
$ python osscmd gs --id=your_id --key=your_key
2013-07-19 08:11 test-oss-sample
Bucket Number is: 1
```

To configure users' ID and Key to the default files, run the following commands. The default oss host is oss.aliyuncs.com.

```
$python osscmd config --id=YOUR_ID --key=YOUR_KEY
```

If you see a prompt saying "Your configuration is saved into" or similar, it indicates the ID and Key have been saved successfully.

Basic operations

List Created Bucket

```
$python osscmd getallbucket
```

The output will be empty if the OSS user didn't create any buckets.

Create Bucket

Create a bucket named mybucketname.

```
$python osscmd createbucket mybucketname
```

Creating a bucket named "mybucketname" may fail because the name of the bucket in OSS is globally unique and someone might have created this bucket. In this case, you need to change the name. For example, you can add a specific date to the bucket name.

Check whether the bucket has been created successfully

```
$python osscmd getallbucket
```

If it fails, check the error message returned.

View Object

After a bucket is successfully created, check the objects in the bucket.

```
$python osscmd list oss://mybucketname/
```

There is no objects in the bucket, so the output is empty.

Upload Object

Upload an object to the bucket. If the local file is named local_existed_file, its MD5 value is shown as below.

```
$ md5sum local_existed_file 7625e1adc3a4b129763d580ca0a78e44 local_existed_file
$ python osscmd put local_existed_file oss://mybucketname/test_object
```

Notice: The md5sum command is used on Linux instead of Windows.

View Object Again

If it is successfully created, check the object again in bucket.

```
$python osscmd list oss://mybucketname/
```

Download Object

Download an object from the bucket to local and compare the md5 value of the file downloaded.

```
$ python osscmd get oss://mybucketname/test_object download_file
$ md5sum download_file
7625e1adc3a4b129763d580ca0a78e44 download_file
```

Notice: The md5sum command is used on Linux instead of Windows.

Delete Object

```
$ python osscmd delete oss://mybucketname/test_object
```

Delete Bucket

Notice: If there are objects in the bucket, the bucket cannot be deleted.

```
$ python osscmd deletebucket mybucketname
```

Usage Lifecycle

Configure an xml text file for lifecycle

```
<LifecycleConfiguration>
<Rule>
<ID>1125</ID>
<Prefix>log_backup/</Prefix>
<Status>Enabled</Status>
<Expiration>
<Days>2</Days>
</Expiration>
</Rule>
</LifecycleConfiguration>
```

This indicates deleting the objects of more than two days old to the current time and with the prefix of log_backup/ in the bucket. For detailed rule configuration, refer to API Reference.

Write Lifecycle

```
python osscmd putlifecycle oss://mybucket lifecycle.xml
0.150(s) elapsed
```

Read Lifecycle

```
python osscmd getlifecycle oss://mybucket
<?xml version="1.0" encoding="UTF-8"?>
<LifecycleConfiguration>
<Rule>
<ID>1125</ID>
<Prefix>log_backup/</Prefix>
<Status>Enabled</Status>
<Expiration>
<Days>2</Days>
</Expiration>
</Rule>
</LifecycleConfiguration>

0.027(s) elapsed
```

Delete Lifecycle

```
python osscmd deletelifecycle oss://mybucket
0.139(s) elapsed
```

Read Lifecycle

```
python osscmd getlifecycle oss://mybucket
Error Headers:
```

```
[('content-length', '288'), ('server', 'AliyunOSS'), ('connection', 'close'), ('x-oss-request-id',
'54C74FEE5D7F6B24E5042630'), ('date', 'Tue, 27 Jan 2015 08:44:30 GMT'), ('content-type', 'application/xml')]
Error Body:

<?xml version="1.0" encoding="UTF-8"?>
<Error>
<BucketName>mybucket</BucketName>
<Code>NoSuchLifecycle</Code>
<Message>No Row found in Lifecycle Table.</Message>
<RequestId>54C74FEE5D7F6B24E5042630</RequestId>
<HostId>mybucket.oss-maque-hz-a.alibaba.net</HostId>
</Error>

Error Status:

404
getlifecycle Failed!
```

Anti-leech Settings

Allow access of blank referer

```
$osscmd putreferer oss://test --allow_empty_referer=true
0.004(s) elapsed
```

Get Configured Referer

```
$osscmd getreferer oss://test
<?xml version="1.0" encoding="UTF-8"?>
<RefererConfiguration>
<AllowEmptyReferer>true</AllowEmptyReferer>
<RefererList />
</RefererConfiguration>
```

Do not allow blank referer. Only allow test referer requests

```
$osscmd putreferer oss://test --allow_empty_referer=false --referer='www.test.com'
0.092(s) elapsed
```

Get Configured Referer

```
$osscmd getreferer oss://test
<?xml version="1.0" encoding="UTF-8"?>
<RefererConfiguration>
<AllowEmptyReferer>false</AllowEmptyReferer>
<RefererList>
<Referer>www.test.com</Referer>
</RefererList>
</RefererConfiguration>
```

Do not allow blank referer. Only allow test and test1 referer requests

```
$osscmd putreferer oss://test --allow_empty_referer=false --referer='www.test.com,www.test1.com'
```

Get Configured Referer

```
$osscmd getreferer oss://test
<?xml version="1.0" encoding="UTF-8"?>
<RefererConfiguration>
<AllowEmptyReferer>false</AllowEmptyReferer>
<RefererList>
<Referer>www.test.com</Referer>
<Referer>www.test1.com</Referer>
</RefererList>
</RefererConfiguration>
```

Use logging

Set logging

```
$osscmd putlogging oss://mybucket oss://myloggingbucket/mb
```

Get logging

```
$osscmd getlogging oss://mybucket
```

Bucket commands

config

Command instructions:

config --id=[accessid] --key=[accesskey] --host=[host] --sts_token=[sts_token]

Configure the default host, ID and Key of the osscmd. The default host is oss.aliyuncs.com.To access oss-internal.aliyuncs.com, you can add —host=oss-internal.aliyuncs.com.The sts_token parameter is not requisite. When sts_token is filled, the tool will perform authentication in STS method.

Example:

- python osscmd config --id=your_id --key=your_key
- python osscmd config --id=your_id --key=your_key --host=oss-internal.aliyuncs.com

getallbucket(gs)

Command instructions:

getallbucket(gs)

Show the bucket the user has created. The gs is the short form of get service. The gs achieves the same effect with getallbucket.

Example:

- python osscmd getallbucket
- python osscmd gs

createbucket(cb,mb,pb)

Command instructions:

createbucket(cb,mb,pb) oss://bucket --acl=[acl]

Create bucket commands. The cb is short for create bucket, mb is short for make bucket, pb is short for put bucket and oss://bucket indicates the bucket. The —acl parameter can be included but it is not required. The several commands all achieve the same effect.

Example:

- python osscmd createbucket oss://mybucket
- python osscmd cb oss://myfirstbucket --acl=public-read
- python osscmd mb oss://mysecondbucket --acl=private
- python osscmd pb oss://mythirdbucket

deletebucket(db)

Command instructions:

deletebucket(db) oss://bucket

Delete bucket commands. The db is short for delete bucket. Deletebucket achieves the same effect with db.

Example:

- python osscmd deletebucket oss://mybucket
- python osscmd db oss://myfirstbucket

deletewholebucket

> **Note:** This command is very risky as it will erase all the data and the erased data cannot be restored. Use it with caution.

Command instructions:

deletewholebucket oss://bucket

Delete bucket and its objects as well as the multipart contents.

Example:

- python osscmd deletewholebucket oss://mybucket

getacl

Command instructions:

getacl oss://bucket

Get bucket access and control privilege.

Example:

- python osscmd getacl oss://mybucket

setacl

Command instructions:

setacl oss://bucket --acl=[acl]

Modify bucket access and control privilege. The acl can only be one of the three, private, public-read, or public-read-write.

Example:

- python osscmd setacl oss://mybucket --acl=private

putlifecycle

Command instructions:

putlifecycle oss://mybucket lifecycle.xml

Set lifecycle rules. The lifecycle.xml is the configuration file of lifecycle. For detailed rule configuration, refer to API Reference.

Example:

- python osscmd putlifecycle oss://mybucket lifecycle.xml

The lifecycle.xml contains the configuration rules of lifecycle. E.g.:

```
<LifecycleConfiguration>
<Rule>
<ID>1125</ID>
<Prefix>log_backup/</Prefix>
<Status>Enabled</Status>
<Expiration>
<Days>2</Days>
</Expiration>
</Rule>
</LifecycleConfiguration>
```

getlifecycle

Command instructions:

osscmd getlifecycle oss://bucket

Get rules of the bucket lifecycle.

Example:

- python osscmd getlifecycle oss://mybucket

deletelifecycle

Command instructions:

osscmd deletelifecycle oss://bucket

Delete all the lifecycle rules under the bucket.

Example:

- python osscmd deletelifecycle oss://mybucket

putreferer

Command instructions:

osscmd putreferer oss://bucket --allow_empty_referer=[true|false] --referer=[referer]

Set anti-leech rules. The allow_empty_referer parameter is requisite and used to set whether it is allowed to be null. The referer parameter is used to set the allowed white list for access, e.g., “www.test1.com,www.test2.com”, with “,” as the separator. For detailed rule configuration, refer to Product documentation.

Example:

- python osscmd putreferer oss://mybucket --allow_empty_referer=true --referer="www.test1.com,www.test2.com"

getreferer

Command instructions:

osscmd getreferer oss://bucket

Get the anti-leech rules of the bucket.

Example:

- python osscmd getreferer oss://mybucket

putlogging

Command instructions:

osscmd putlogging oss://source_bucket oss://target_bucket/[prefix]

The source_bucket indicates the bucket for logs, and the target_bucket indicates where the logs can be stored. You can set a prefix for the log files generated in the source bucket for the convenience of categorized query.

Example:

- python osscmd putlogging oss://mybucket oss://myloggingbucket/mb

getlogging

Command instructions:

osscmd getlogging oss://bucket

Get the logging rules of the bucket and an xml file will be returned.

Example:

- python osscmd getlogging oss://mybucket

Object commands

ls(list)

Command instructions:

ls(list) oss://bucket/[prefix] [marker] [delimiter] [maxkeys]

List object in the bucket.

Example:

- python osscmd ls oss://mybucket/folder1/folder2
- python osscmd ls oss://mybucket/folder1/folder2 maker1
- python osscmd ls oss://mybucket/folder1/folder2 maker1 /
- python osscmd ls oss://mybucket/
- python osscmd list oss://mybucket/ "" "" 100

Command instructions:

ls(list) oss://bucket/[prefix] --marker=xxx --delimiter=xxx --maxkeys=xxx

List object in the bucket.

Example:

- python osscmd ls oss://mybucket/folder1/folder2 --delimiter=/
- python osscmd ls oss://mybucket/folder1/folder2 --maker=a
- python osscmd ls oss://mybucket/folder1/folder2 --maxkeys=10

mkdir

Command instructions:

mkdir oss://bucket/dirname

Create an object ending with “/” of a size of 0.

Example:

- python osscmd mkdir oss://mybucket/folder

listallobject

Command instructions:

listallobject oss://bucket/[prefix]

Show all objects in the bucket, and the prefix can be specified.

Example:

- python osscmd listallobject oss://mybucket
- python osscmd listallobject oss://mybucket/testfolder/

deleteallobject

Command instructions:

deleteallobject oss://bucket/[prefix]

Delete all objects in the bucket, and the prefix can be specified.

Example:

- python osscmd deleteallobject oss://mybucket
- python osscmd deleteallobject oss://mybucket/testfolder/

downloadallobject

Command instructions:

downloadallobject oss://bucket/[prefix] localdir --replace=false --thread_num=5

Download the objects in the bucket to a local directory, with the directory structure unchanged. The prefix can be specified for downloading. —replace=false indicates that if a local file already exists with the same name, it will not be replaced during the download. —replace=true indicates that the local file with the same name will be replaced. The thread_num can be used to configure the download threading.

Example:

- python osscmd downloadallobject oss://mybucket /tmp/folder
- python osscmd downloadallobject oss://mybucket /tmp/folder –-replace=false
- python osscmd downloadallobject oss://mybucket /tmp/folder –-replace=true --thread_num=5

downloadtodir

Command instructions:

downloadallobject oss://bucket/[prefix] localdir --replace=false

Download the objects in the bucket to a local directory, with the directory structure unchanged. The prefix can be specified for downloading. —replace=false indicates that if a local file already exists with the same name, it will not be replaced during the download. —replace=true indicates that the local file with the same name will be replaced. It achieves the same effect with the downloadallobject.

Example:

- python osscmd downloadtodir oss://mybucket /tmp/folder
- python osscmd downloadtodir oss://mybucket /tmp/folder –-replace=false
- python osscmd downloadtodir oss://mybucket /tmp/folder –-replace=true

uploadfromdir

Command instructions:

uploadfromdir localdir oss://bucket/[prefix] --check_point=check_point_file --replace=false --check_md5=false --thread_num=5

Upload local files into the bucket. E.g., the localdir is /tmp/

There are three files a/b, a/c, and a, and they will be oss://bucket/a/b, oss://bucket/a/c, oss://bucket/a after being uploaded into the OSS. If the prefix is specified as mytest, the uploaded files to OSS will be oss://bucket/mytest/a/b, oss://bucket/mytest/a/c, and oss://bucket/mytest/a.

--check_point=check_point_file is the specified file. After the files are specified, osscmd will put the uploaded local files into check_point_file as time stamps, and the uploadfromdir command will compare the time stamps of the files being uploaded with that recorded in check_point_file. If there are changes, the files will be re-uploaded. Otherwise the file will be skipped. The check_point_file does not exist by default. --replace=false indicates that if a local file already exists with the same name, it will not be replaced during the download. —replace=true indicates that the local file with the same name will be replaced. --check_md5=false indicates that when the files are being uploaded, the Content-MD5 request header will not undergo verification. True indicates that the Content-MD5 request header will undergo verification.

> Note: the logs in the check_point_file involve all the uploaded files. When there are too many files uploaded, the check_point_file will be sizable.

Example:

- python osscmd uploadfromdir /mytemp/folder oss://mybucket
- python osscmd uploadfromdir /mytemp/folder oss://mybucket --check_point_file=/tmp/mytemp_record.txt
- python osscmd uploadfromdir C:\Documents and Settings\User\My Documents\Downloads oss://mybucket --check_point_file=C:\cp.txt

put

Command instructions:

put localfile oss://bucket/object --content-type=[content_type] --headers="key1:value1#key2:value2" --check_md5=false

When uploading a local file into the bucket, you can specify the object content-type, or specify

customized headers. --check_md5=false indicates that when the files are being uploaded, the Content-MD5 request header will not undergo verification. True indicates that the Content-MD5 request header will undergo verification.

Example:

- python osscmd put myfile.txt oss://mybucket
- python osscmd put myfile.txt oss://mybucket/myobject.txt
- python osscmd put myfile.txt oss://mybucket/test.txt --content-type=plain/text --headers= "x-oss-meta-des:test#x-oss-meta-location:CN"
- python osscmd put myfile.txt oss://mybucket/test.txt --content-type=plain/text

upload

Command instructions:

upload localfile oss://bucket/object --content-type=[content_type] --check_md5=false

Upload local files in object group. Not recommended. --check_md5=false indicates that when the files are being uploaded, the Content-MD5 request header will not undergo verification. True indicates that the Content-MD5 request header will undergo verification.

Example:

- python osscmd upload myfile.txt oss://mybucket/test.txt --content-type=plain/text

get

Command Instructions:

get oss://bucket/object localfile

Download the object to local.

Example:

- python osscmd get oss://mybucket/myobject /tmp/localfile

multiget(multi_get)

Command instructions:

multiget(multi_get) oss://bucket/object localfile --thread_num=5

Download the object to local in multithreading. The thread count can be configured.

Example:

- python osscmd multiget oss://mybucket/myobject /tmp/localfile

- python osscmd multi_get oss://mybucket/myobject /tmp/localfile

cat

Command instructions:

cat oss://bucket/object

Read object content and print them out directly. Do not use it when the object content is big in size.

Example:

- python osscmd cat oss://mybucket/myobject

meta

Command instructions:

meta oss://bucket/object

Read the meta information of the object and print it out. The meta information includes the content-type, file length, custom meta, etc.

Example:

- python osscmd meta oss://mybucket/myobject

copy

Command instructions:

copy oss://source_bucket/source_object oss://target_bucket/target_object --headers="key1:value1#key2:value2"

Copy the source object of the source bucket to the destination object in the destination bucket.

Example:

- python osscmd copy oss://bucket1/object1 oss://bucket2/object2

rm(delete,del)

Command instructions:

rm(delete,del) oss://bucket/object

Delete object.

Example:

- python osscmd rm oss://mybucket/myobject
- python osscmd delete oss://mybucket/myobject
- python osscmd del oss://mybucket/myobject

signurl(sign)

Command instructions:

signurl(sign) oss://bucket/object --timeout=[timeout_seconds]

Generate a URL containing the signature and specify the timeout value. This is applicable to the scenario where the private bucket provides the specified object for others' accesses.

Example:

- python osscmd sign oss://mybucket/myobject
- python osscmd signurl oss://mybucket/myobject

Mutipart commands

init

Command instructions:

init oss://bucket/object

Initiate and generate an Upload ID. The Upload ID can be used in combination with the multiupload command.

Example:

- python osscmd init oss://mybucket/myobject

listpart

Command instructions:

listpart oss://bucket/object --upload_id=xxx

Show the uploaded parts of an Upload ID in the designated object. See OSS API Reference for related concepts. The Upload ID must be designated.

Example:

- python osscmd listpart oss://mybucket/myobject --upload_id=

75835E389EA648C0B93571B6A46023F3

listparts

Command instructions:

listparts oss://bucket

Show the uncompleted multipart Upload ID and objects in the bucket. When you want to delete a bucket but system prompts that the bucket is not empty, this command can be used to check whether there are multipart contents.

Example:

- python osscmd listparts oss://mybucket

getallpartsize

Command instructions:

getallpartsize oss://bucket

Show the total size of parts of the existing Upload ID in the bucket.

Example:

- python osscmd getallpartsize oss://mybucket

cancel

Command instructions:

cancel oss://bucket/object --upload_id=xxx

Terminate the Multipart Upload event of the Upload ID.

Example:

- python osscmd cancel oss://mybucket/myobject --upload_id=
  D9D278DB6F8845E9AFE797DD235DC576

multiupload(multi_upload,mp)

Command instructions:

multiupload(multi_upload,mp) localfile oss://bucket/object --check_md5=false --thread_num=10

Upload local files to the OSS by multipart.

Example:

- python osscmd multiupload /tmp/localfile.txt oss://mybucket/object
- python osscmd multiup_load /tmp/localfile.txt oss://mybucket/object
- python osscmd mp /tmp/localfile.txt oss://mybucket/object

Command instructions:

multiupload(multi_upload,mp) localfile oss://bucket/object --upload_id=xxx --thread_num=10 --max_part_num=1000 --check_md5=false

Upload local files to the OSS by multipart. The part count of the local file is defined in max_part_num. This command will first judge whether the ETag of corresponding parts of the Upload ID is consistent with the MD5 value of the local file. If yes, the upload will be skipped. So if an Upload ID is generated before use, it will be included as a parameter. Even if the upload fails, it can be resumed by repeating the multiupload command. --check_md5=false indicates that when the files are being uploaded, the Content-MD5 request header will not undergo verification. True indicates that the Content-MD5 request header will undergo verification.

Example:

- python osscmd multiupload /tmp/localfile.txt oss://mybucket/object --upload_id=D9D278DB6F8845E9AFE797DD235DC576
- python osscmd multiup_load /tmp/localfile.txt oss://mybucket/object --thread_num=5
- python osscmd mp /tmp/localfile.txt oss://mybucket/object --max_part_num=100

copylargefile

Command instructions:

copylargefile oss://source_bucket/source_object oss://target_bucket/target_object --part_size=10*1024*1024 --upload_id=xxx

When copying a large file of over 1G, the object can be copied to the destination location through multipart (The source bucket and the destination bucket must be in the same region). The upload_id is an optional parameter. If you need to resume the transmission of a multipart copy event, you can include the upload_id. The part_size is used to define the part size. A single part should be 100KB at minimal, and up to 10,000 parts are supported. If the set value of part_size conflicts with the OSS limit, the application will automatically adjust the part size.

Example:

- python osscmd copylargefile oss://source_bucket/source_object oss://target_bucket/target_object --part_size=10*1024*1024

uploadpartfromfile (upff)

Command instructions:

uploadpartfromfile (upff) localfile oss://bucket/object --upload_id=xxx --part_number=xxx

This command is mainly used for test and not recommended for actual use.

uploadpartfromstring(upfs)

Command instructions:

uploadpartfromstring(upfs) oss://bucket/object --upload_id=xxx --part_number=xxx --data=xxx

This command is mainly used for test and not recommended for actual use.

ossprobe

Introduction

The ossprobe is an OSS access detection tool used to troubleshoot problems caused by network errors or incorrect settings of basic parameters during the upload and download processes. If an error occurs after you run a command to upload or download data, the ossprobe displays the possible cause to help you identify the error quickly.

Version

Version: 1.0.0

Key functions

- Checks whether the network environment is normal
- Checks whether basic parameters are correct
- Tests the upload and download speeds

Platforms

- Linux
- Windows
- Mac

Download software

- windows64 ossprobe

- linux64 ossprobe
- mac ossprobe

Detect download problems

Usage

```
    ossprobe --download  [-i AccessKeyId] [-k AccessKeySecret] [-p EndPoint] [-b BucketName] [-o ObjectName] [-t
LocalPath]
[-f Url] [-a Address]

-f --from Object的Url
-i --id AccessKeyId
-k --key AccessKeySecret
-p --endpoint EndPoint
-b --bucket BucketName
-o --object ObjectName
-t --to Save path for the downloaded content. By default, it is the path to a temporary file in the current directory.
-a --addr Network address for detection. The default address is www.aliyun.com. If you are using private cloud,
select an accessible address in the private cloud.

TIP: If the -f parameter is present, a URL is used for download. If the -f parameter is not present, you must set the
AccessKeyID, AccessKeySecret, EndPoint, and BucketName parameters.
```

Example

To check whether URL-based download is normal (How to obtain a URL), run the following commands:

Method	Command
Download from a specified URL	ossprobe --download -f Url
Download from a specified URL and save the downloaded content to a specified file	ossprobe --download -f Url -t tmp/example.txt
Download from a specified URL and detect the network condition of a specified address	ossprobe --download -f Url -a Addr

To check whether download using specified parameters (AccessKeyID, AccessKeySecret, EndPoint, and BucketName) is normal, run the following commands:

Method	Command
Download a random file	ossprobe --download -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName
Download a specified file	ossprobe --download -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName -o ObjectName
Download a specified file and save the	ossprobe --download -i AccessKeyId -k

downloaded content to a specified local file	AccessKeySecret -p EndPoint -b BucketName -o ObjectName -t tmp/example.txt
Download a random file and detect the network condition of a specified address	ossprobe --download -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName -a Addr

TIP:

- The file you downloaded is a binary executable program, and you need to add the ossprobe executable permissions through chmod +x ossprobe in the Linux system.
- By default, the -t parameter indicates the path to a temporary file in the current directory (the file name format is ossfilestore20160315060101).
- If the -t parameter indicates a directory, a temporary file is generated in the directory to save data (the file name format is ossfilestore20160315060101).
- If a file is downloaded from a URL, the file is named after the last string following the forward slash "/ " in the URL. For example, if the URL is http://aliyun.com/a.jpg, then the file is saved as a.jpg.

Detect upload problems

Usage

```
ossprobe --upload -i AccessKeyId -k AccessKeySecret -p EndPoint  -b BucketName [-m normal|append|multipart] [-s UploadFilePath] [-o ObjectName] [-a Addr]

-i --id AccessKeyID
-k --key AccessKeySecret
-p --endpoint EndPoint
-b --bucket BucketName
-s --src Path to the file you want to upload. By default, it is the path to a local temporary file.
-m --mode File upload mode. The default is normal upload.
-o --object Uploaded object name. By default, the object name is the name of the uploaded file if -s is not null. If -s is null, by default, the object name is the name of the temporary file starting with tem.
-a --addr Network address for detection. The default address is the address of the Alibaba Cloud website. If you are using private cloud, select an accessible address in the private cloud.
```

Example

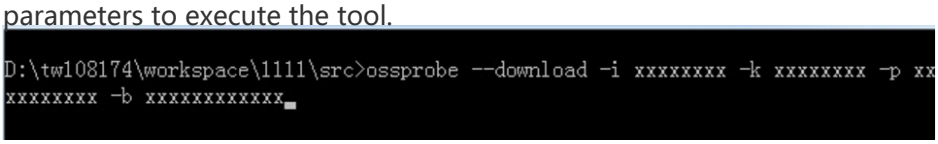
Method	Command
Generate a temporary file and upload it in normal mode	ossprobe --upload -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName
Generate a temporary file and upload it in append mode	ossprobe --upload -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName -o ObjectName -m append
Generate a temporary file and upload it in multipart mode	ossprobe --upload -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName

	-o ObjectName -m multipart
Upload specified content in multipart mode	ossprobe --upload -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName -o ObjectName -m multipart -s src
Upload specified content in multipart mode and specify the object name	ossprobe --upload -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName -m multipart -s src -o example.txt
Generate a temporary file, upload it in normal mode, and detect the network condition of a specified address	ossprobe --upload -i AccessKeyId -k AccessKeySecret -p EndPoint -b BucketName -a Addr

TIP: The name of a randomly generated file starts with ossuploadtmp.

Platform differences

- Windows
  Press Win+R to bring up the "Run" dialog box, enter cmd, and press Enter. On the command-line interface (CLI), enter the path to the tool and fill in related detection parameters to execute the tool.



- Linux and Mac
  Open the terminal. On the displayed interface, enter the path to the tool and fill in related detection parameters to execute the tool.

View report data

After command execution, a report named logOssProbe20060102150405.txt is generated (the numbers following logOssProbe indicate the formatted date of report generation). The possible error cause is printed in command line mode. If you think the error message is not specific, you can view the report. If the problem persists, you can submit a ticket attached with the detection report.

Console display

The console displays the following main information:

- After execution, the steps marked with × fail, whereas the steps not marked with × are successful.
- The result indicates whether the upload or download operation is successful. If the upload or

download operation is successful, the console displays the file size and upload/download time.
- The “Suggested Change” column shows the error cause or change suggestions.
- If you are familiar with OSS error codes, you can perform troubleshooting based on the error message returned by OSS.
- The “Log Info” columns shows the log name and address, allowing you to find the log.

(TIP: No change suggestions may be given when an error is detected. When this happens, perform troubleshooting based on the returned error code by referring to OSS error code.)

Log file

Different from console display, log files contain network detection details. Ping is used to detect a specified network or the network of a specified EndPoint, tracert is used to detect the route for EndPoint access, and nslookup is used for DNS detection.

References

OSS error code

Naming conventions of buckets and objects

How to obtain a URL

Official migration tool

Overview

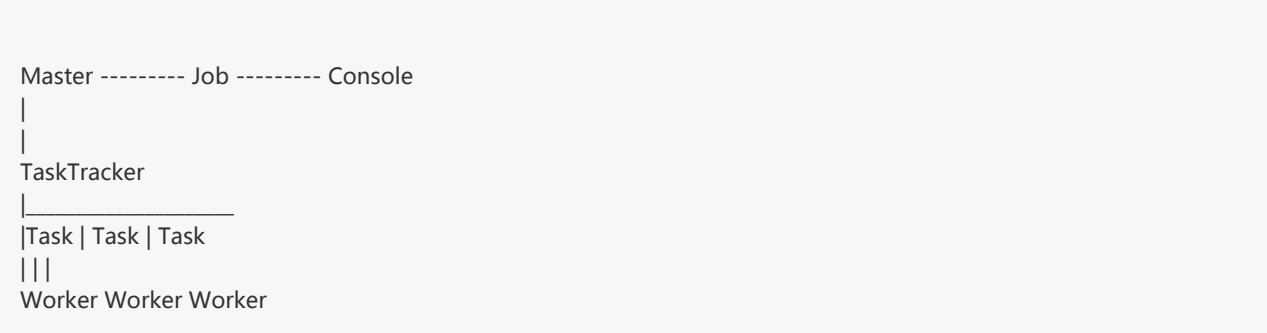
The OssImport tool allows you to migrate data stored locally or in other cloud storage systems to the OSS. It has the following features:

- Support a rich variety of data sources including local drives, Qiniu, Baidu BOS, AWS S3, Azure Blob, Youpai Cloud, Tencent Cloud COS, Kingsoft KS3, HTTP, and OSS, and can be expanded as needed.
- Support resumable data transfers.
- Support traffic control.
- Support migrating objects after a specified time point or with a specified prefix.

- Support parallel data uploads and downloads.
- Support standalone and distributed modes. The standalone mode is easy to deploy and use, and the distributed mode is suitable for large-scale data migration.

Architecture

The OssImport is based on the **master-worker** distributed architecture, as shown in the figure below:



Job: The data migration jobs submitted by users. For users, one job corresponds to one configuration file **job.cfg**.

Task: A job can be divided into multiple tasks by data size and number of files. Each task migrates a portion of files. The minimal unit for dividing a job into tasks is a file. One file cannot be split into multiple tasks.

The OssImport tool modules are listed in the following table:

Role	Description
Master	The master is responsible for splitting a job into multiple tasks by data size and number of files. The data size and number of files can be configured in **sys.properties**. The detailed process for splitting a job into multiple tasks is as follows: - The master node scans the full list of files to be migrated from the local/other cloud storage devices. - The master splits the full file list into tasks by data size and the number of files and each task is responsible for the migration or validation for a part of files.
Worker	- The worker is responsible for file migration and data validation of tasks. It pulls the specific file from the data source and uploads the file to the specified directory to the OSS. You can specify the data source to be migrated and the OSS configuration in **job.cfg** or **local_job.cfg**.

	- Worker data migration supports limiting traffic and specifying the number of concurrent tasks. You can configure the settings in **sys.properties**.
TaskTracker	TaskTracker is abbreviated to Tracker. It is responsible for distributing tasks and tracking task statuses.
Console	The console is responsible for interacting with users and receiving command display results. It supports system management commands such as **deploy**, **start**, and **stop**, and job management commands such as **submit**, **retry**, and **clean**.

Deployment modes

The OssImport has two deployment modes available: **standalone mode** and **distributed mode**. The standalone mode is sufficient for small-scale data migration with data smaller than **30 TB**. Distributed mode is recommended for larger data migrations.

Standalone

The master, worker, tracker, and console run on the same machine. There is only one worker in the system. We have encapsulated and optimized the deployment and execution of the standalone mode and the standalone deployment and execution are both very easy. In standalone mode, the master, worker, tasktracker, and console modules are packaged into **ossimport2.jar**.

The file structure in standalone mode is as follows:

```
ossimport
├── bin
│ └── ossimport2.jar # The JAR including master, worker, tracker, and console modules
├── conf
│ ├── local_job.cfg # Standalone job configuration file
│ └── sys.properties # Configuration file for the system running parameters
├── console.bat # Windows command line, which can execute distributed call-in tasks
├── console.sh # Linux command line, which can execute distributed call-in tasks
├── import.bat # The configuration file for one-click import and execution in Windows is the data migration job configured in conf/local_job.cfg, including start, migration, validation, and retry
├── import.sh # The configuration file of one-click import and execution in Linux is the data migration job configured in conf/local_job.cfg, including start, migration, validation, and retry
├── logs # Log directory
└── README.md # Description documentation. We recommend that you carefully read the documentation before using this feature
```

Notice:

- The **import.bat** or **import.sh** file is a one-click import script and can be run directly after you complete modification to **local_job.cfg**.
- The **console.bat** or **console.sh** is the command line tool and can be used for distributed

execution of commands.
- Execute scripts or commands in the ossimport directory, that is, the directory at the same level as *.bat/*.sh.

Distributed

In distributed mode, you can start multiple worker nodes for data migration. Tasks are evenly allocated to the worker nodes and one worker node can execute multiple tasks. One machine can only start one worker node. The master will be started at the same time as the first worker node configured in workers, and the tasktracker and console will also run on the machine.

The file structure in distributed mode is as follows:

```
ossimport
├── bin
│ ├── console.jar # The JAR package of the console module
│ ├── master.jar # The JAR package of the master module
│ ├── tracker.jar # The JAR package of the tracker module
│ └── worker.jar # The JAR package of the worker module
├── conf
│ ├── job.cfg # The template of the job configuration file
│ ├── sys.properties # Configuration file for the system running parameters
│ └── workers # Worker list
├── console.sh # The command line tool. Currently it only supports Linux
├── logs # Log directory
└── README.md # Description documentation. We recommend that you carefully read the documentation before using the feature
```

Notice: The distributed command line tool **console.sh** currently only supports Linux and does not support Windows.

Configuration files

In standalone mode, there are two configuration files: **sys.properties** and **local_job.cfg**. In distributed mode, there are three configuration files: **sys.properties**, **local_job.cfg**, and **workers**. Specifically, **local_job.cfg** and **job.cfg** are identical, except in name. The **workers** file is exclusive to the distributed environment.

sys.properties

System running parameters.

Field	Meaning	Description
workingDir	Working directory.	- The directory after the tool kit is extracted.

		- Do not modify this option in standalone mode. - The working directory of each machine must be the same in distributed mode.
workerUser	The worker machine SSH user name.	- If you have configured privateKeyFile, the privateKeyFile is used in priority. - If privateKeyFile is not configured, the workerUser/workerPassword combination is used. - Do not modify this option in standalone mode.
workerPassword	The worker machine SSH user password.	Do not modify this option in standalone mode.
privateKeyFile	The file path of the public key.	- If you have established an SSH channel, you can specify the public key file path. Otherwise, leave it empty. - If you have configured privateKeyFile, the privateKeyFile is given priority. - If privateKeyFile is not configured, the workerUser/workerPassword is used. - Do not modify this option in the standalone mode.
sshPort	The SSH port.	The default value is 22. It does not usually need to be changed. Do not modify this option in standalone mode.
workerTaskThreadNum	The maximum number of threads for the worker to execute tasks.	- This parameter is related to the machine memory and network. Recommended value is 60. - The value can be increased, for example to 150 for physical machines. If the network bandwidth is already full, do not increase the value further. - If the network is poor, lower the value as

		appropriate to, for example, 30. This way, you can avoid the timeout of a large number of requests from request competition.
workerMaxThroughput(KB/s)	The data migration traffic ceiling on the worker node.	This value limits the traffic. The default value 0 indicates that no traffic limitations are imposed.
dispatcherThreadNum	The number of threads for task distribution and status confirmation of the tracker.	The default value should be enough. You don’ t need to change the default value if you have no special requirements.
workerAbortWhenUncatchedException	Whether to skip or abort in case of an unknown error.	Unknown errors are skipped by default.
workerRecordMd5	Whether to use metadata **x-oss-meta-md5** to log the MD5 value of the migrated file in the OSS. The default setting is no.	It is mainly used for file data validation using MD5.

job.cfg

Data migration job configuration. The **local_job.cfg** and **job.cfg** options are identical except in name.

Field	Meaning	Description
jobName	The job name, a string.	- The unique identifier of the job. The naming rule is [a-zA-Z0-9_-]{4,128}. It supports the submission of multiple jobs of different names. - If you submit a job with the same name as another job, the system will prompt that the job already exists. You are not allowed to submit a job of the same name before you clean the original job with the name.
jobType	The job type, a string.	There are two types: **import** and **audit**. The default value is **import**. - import: Execute the data migration and validate the migrated data for consistency. - audit: Only validate data consistency.
isIncremental	Whether to enable	- Default value: False.

	incremental migration mode, a Boolean value.	- If it is set to true, incremental data is rescanned at the interval specified by incrementalModeInterval (unit: second) and synchronized to the OSS.
incrementalModeInterval	Synchronization interval in incremental mode, an integer value. Unit: second.	Valid when isIncremental=true. The minimum configurable interval is 900 seconds. We do not recommend you configure it to a value smaller than 3,600 seconds as that will waste a large number of requests and lead to additional system overhead.
importSince	Migrate data later than this time value, an integer value. Unit: second.	- This time value is a Unix timestamp, that is, the number of seconds since UTC00:00 on January 1, 1970. You can get the value through the date +%s command. - The default value is 0, indicating to migrate all the data.
srcType	The synchronization source type, a string. Case sensitive.	Currently this parameter supports 10 types including local, oss, qiniu, bos, ks3, s3, youpai, HTTP, cos, and azure. - local: Migrate data from a local file to the OSS. You only need to fill in the srcPrefix for this option and do not need to fill in srcAccessKey, srcSecretKey, srcDomain, and srcBucket. - oss: Migrate data from one bucket to another. - qiniu: Migrate data from Qiniu cloud storage to the OSS. - bos: Migrate data from Baidu cloud storage to the OSS. - ks3: Migrate data from Kingsoft cloud storage to the OSS. - s3: Migrate data from AWS S3 to the OSS. - youpai: Migrate data from Youpai Cloud to the OSS. - HTTP: Migrate data to the OSS through the provided HTTP link

		list. - cos: Migrate data from the Tencent cloud storage COS to the OSS. - azure: Migrate data from Azure Blob to the OSS.
srcAccessKey	The source AccessKey, a string.	Fill in the AccessKey of the data source if srcType is set to oss, qiniu, baidu, ks3 or s3. - or the local and HTTP types, this option can be left empty. - For youpai and azure types, fill in the AccountName.
srcSecretKey	The source SecretKey, a string.	Fill in the SecretKey of the data source if srcType is set to oss, qiniu, baidu, ks3 or s3. - For the local and HTTP types, this option can be left empty. - youpai: Fill in the operator password. - azure: Fill in the AccountKey.
srcDomain	Source endpoint.	This configuration item is not required if the srcType is set to local or HTTP. - oss: The domain name obtained from the console. It is a second-level domain name without the bucket prefix. A full list can be found at domain name list. - qiniu: The domain name of the corresponding bucket obtained from the Qiniu console. - bos: The Baidu BOS domain name, such as http://bj.bcebos.com or http://gz.bcebos.com. - ks3: Kingsoft KS3 domain name, such as http://kss.ksyun.com, http://ks3-cn-beijing.ksyun.com or http://ks3-us-west-1.ksyun.coms. - The S3 and AWS S3 domain names of various regions can be

		found at S3 Endpoint. - youpai: The domain name of the Youpai Cloud, such as automatic identification of the optimal channel of http://v0.api.upyun.com, or telecommunication line http://v1.api.upyun.com, or China Unicom or China Netcom line http://v2.api.upyun.com or China Mobile or China Railcom line http://v3.api.upyun.com. - cos: The bucket region of the Tencent Cloud, such as South China: gz, North China: tj, and East China: sh. - azure: The EndpointSuffix in the Azure Blob connection string, such as core.chinacloudapi.cn.
srcBucket	The name of the source bucket or the container.	This configuration item is not required if the srcType is set to local or HTTP. azure: Fill in the container name in Azure Blob, and fill in the bucket name for others.
srcPrefix	The source prefix, a string. The default value is empty.	If the srcType is set to **local**, fill in the local directory in full, separated and ended by **/**, such as **c:/example/** or **/data/example/**. If the srcType is oss, qiniu, bos, ks3, youpai or s3, the value is the prefix of the object to be synchronized, without the bucket name, such as **data/to/oss/**. If you want to synchronize all the objects, leave the srcPrefix empty.
destAccessKey	The destination AccessKey, a string.	To view the OSS AccessKeyID, log on to the console.
destSecretKey	The destination SecretKey, a string.	To view the OSS AccessKeySecret, log on to the console.
destDomain	Destination endpoint, a string.	Obtained from the console. It is a second-level domain name without the bucket prefix. A full list can be found at domain name list.
destBucket	The destination bucket, a	The OSS bucket name. It does

	string.	not need to end with /.
destPrefix	The destination prefix, a string. The default value is empty.	- The destination prefix. The default value is empty in which case the objects are placed in the destination bucket. - If you want to synchronize data to a specific directory on the OSS, end the prefix with /, such as **data/in/oss/**. -Note that the OSS does not support / as the object prefix, so do not set destPrefix to start with /. - A local file in the path **srcPrefix+relativePath** will be migrated to **destDomain/destBucket/destPrefix + relativePath** on the OSS. - An object on the cloud in the path **srcDomain/srcBucket/srcPrefix+relativePath** will be migrated to **destDomain/destBucket/destPrefix + relativePath** on the OSS.
taskObjectCountLimit	The maximum number of files in a task, an integer. The default value is 10,000.	This configuration option will affect the concurrency of the executed jobs. Generally the configuration is set to the total number of files/total number of workers/number of migration threads (workerTaskThreadNum) and the maximum number is 50,000. If the total number of files is unknown, use the default value.
taskObjectSizeLimit	The maximum data size in a task, an integer. Unit: bytes. The default value is 1 GB.	This configuration option will affect the concurrency of the executed jobs. Generally the configuration is set to the total data size/total number of workers/number of migration threads (workerTaskThreadNum). If the total data size is unknown, use the default value.
scanThreadCount	The number of threads for parallel file scanning, an integer. The default value is 1.	This configuration option is related to file scanning efficiency. Do not modify the configuration if you have no special requirements.
maxMultiThreadScanDepth	The maximum allowable depth of directories for the	- The default value of 1 indicates parallel scan on top-level

	parallel scan, an integer. The default value is 1.	directories. - Do not modify this configuration if you have no special requirements. If the value is configured too large, the job may fail to run normally.
appId	The appId of the Tencent COS, an integer.	Valid when srcType is set to **cos**.
httpListFilePath	The absolute path of the HTTP list file, a string.	- Valid when srcType is set to HTTP. When the source is an HTTP link address, you are required to provide the absolute path of the file with the HTTP link address as the content, such as **c:/example/http.list**. - The HTTP link in the file should be divided into two columns separated by spaces, representing the prefix and the relative path on the OSS after the upload respectively, such as **c:/example/http.list** which contains the following content: http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/aa/bb.jpg http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/cc/dd.jpg The object names for the two rows after they are migrated to the OSS are **destPrefix + bb.jpg** and **destPrefix + cc/dd.jpg** respectively.

Workers

The workers is exclusive to the distributed mode and every IP address is a row, such as:

```
192.168.1.6
192.168.1.7
192.168.1.8
```

Notice:

- In the preceding configuration, the master, worker, and TaskTracker will be started on 192.168.1.6 and the console also needs to be executed on the machine.
- Make sure that the user name, logon mode, and working directory of multiple worker modes are the same.

Running environment

Java 1.7 and later.

Download

Standalone deployment supports Linux and Windows.

Download the tool for standalone deployment: ossimport-2.2.1.zip.

Download the tool to a local directory and use the tool or the unzip command to unzip the files. The file structure after unzipping is as follows:

```
ossimport
├── bin
│ └── ossimport2.jar # The JAR including master, worker, tracker and console modules
├── conf
│ ├── local_job.cfg # The job configuration file
│ └── sys.properties # Configuration file of the system running parameters
├── console.bat # Windows command line, which can execute distributed call-in tasks
├── console.sh # Linux command line, which can execute distributed call-in tasks
├── import.bat # The configuration file for one-click import and execution in Windows is the data migration job configured in conf/local_job.cfg, including start, migration, validation and retry
├── import.sh # The configuration file for one-click import and execution in Linux is the data migration job configured in conf/local_job.cfg, including start, migration, validation and retry
├── logs # Log directory
└── README.md # Description documentation. We strongly recommend you carefully read the documentation before using the feature
```

Configuration

The standalone version has two configuration files: conf/sys.properties and conf/local_job.cfg. For details on the configuration items, see the Introduction chapter.

Do not change the configuration items in conf/sys.properties: workingDir, workerUserName, workerPassword, and privateKeyFile. Do not change the conf/local_job.cfg name and location. Do not change the jobName configuration item. Configure other items appropriately.

> **Notice:** Confirm the parameters in **sys.properties** and **local_job.cfg** before submitting the job. The parameters in the job are not allowed to be changed after the job is submitted.

Running

In standalone mode, a data migration job has two execution modes: one-click import and step-by-step execution.

One-click import encapsulates all the steps and data migration can then be completed following the prompts of the script. **We strongly recommend one-click import for new users.**

Step-by-step execution includes executing the starting service, submitting the job and retrying failed tasks.

The configuration file for standalone mode is **conf/local_job.cfg**. Change the job parameters based on actual needs prior to data migration. Do not change the default job name **local_test**.

One-click import

To execute one-click import, execute import.bat in **cmd.exe** in Windows, and execute bash import.sh in Linux.

If you previously executed this job, you will be asked if you want to continue the job from the last break point or if you want to execute a new synchronization job. If you initiate a new data migration job, or have modified the synchronized source end/destination end, re-execute the synchronization job.

After a job starts in Windows, a new cmd window will pop up showing the synchronization job in progress and the log. The job status in the old window is refreshed every 10 seconds. Do not close these two windows during the data migration process. In Linux, the above process is executed in the background.

When the job is complete, if a task failed, you will be asked if you want to retry. Enter **y** to retry or **n** to skip this step and exit.

To see why the upload failed, open the file master/jobs/local_test/failed_tasks/<tasktaskid>/audit.log and check the cause of the failure.

Step-by-step execution

Clear jobs with the same name. If you have run job with the same name before and want to execute the job again, first clear the job with the same name. If you have never run the job or you want to retry a failed job, do not run the clear command. In Windows, execute console.bat clean in **cmd.exe**. In Linux, execute bash console.sh clean.

Submit the data migration job. The OssImport does not support submitting jobs of the same

name. If there are jobs with the same name, clear the job with the same name first. The configuration file for the submitted job is **conf/local_job.cfg**, and the default job name is **local_test**. To submit a job, execute console.bat submit in **cmd.exe** in Windows, and execute bash console.sh submit in Linux.

Start the service. Execute console.bat start in **cmd.exe** in Windows, and execute bash console.sh start in Linux.

View the job status. Execute console.bat stat in **cmd.exe** in Windows, and execute bash console.sh stat in Linux.

Retry a failed task. Tasks may fail due to network issues or other causes. Only failed tasks will be retried. Execute console.bat retry in **cmd.exe** in Windows, and execute bash console.sh retry in Linux.

Stop the service. Close the window **%JAVA_HOME%/bin/java.exe** in Windows, and execute bash console.sh stop in Linux.

Notice: We recommend that you use one-click import for data migration if you have no special requirements.

Common causes of failure

A file in the source directory was modified during the upload process. This cause will be indicated by a SIZE_NOT_MATCH error in **log/audit.log**. In this case, the old file has been uploaded successfully, but the changes have not been synchronized to the OSS.

A source file was deleted during the upload process, leading to download failure.

A source file name does not conform to naming rules of the OSS (file name cannot start with / or be empty), leading to upload failure.

The data source file failed to be downloaded.

The program exited unexpectedly and the job status is **Abort**. If this happens, contact after-sales technical support.

Job statuses and logs

After a job is submitted, the master splits the job into tasks, the workers execute the tasks and the tracker collects the task statuses. After a job is completed, the ossimport directory contains the following:

```
ossimport
├── bin
│ └── ossimport2.jar # The standalone version JAR
├── conf
│ ├── local_job.cfg # The job configuration file
│ └── sys.properties # Configuration file of the system running parameters
├── console.sh # The command line tool
├── import.sh # One-click import script
├── logs
│ ├── import.log # Archive logs
│ ├── job_stat.log # Job status record
│ ├── ossimport2.log # Running log of the standalone version
│ └── submit.log # Job submission record
├── master
│ ├── jobqueue # Store jobs that have not been fully split
│ └── jobs # Store the job running status
│ └── local_test # Job name
│ ├── checkpoints # The checkpoint record of the master splitting the job to tasks
│ │ └── 0
│ │ └── 034DC9DD2860B0CFE884242BC6FF92E7.cpt
│ ├── dispatched # Tasks that have been assigned to the workers but haven't been fully run
│ │ └── localhost
│ ├── failed_tasks # Tasks that failed to run
│ ├── pending_tasks # Tasks that have not been assigned
│ └── succeed_tasks # Tasks that run successfully
│ └── A41506C07BF1DF2A3EDB4CE31756B93F_1499744514501@localhost
│ ├── audit.log # The task running log. You can view the error causes in the log
│ ├── DONE # Mark of successful tasks
│ ├── error.list # The task error list. You can view the error file list
│ ├── STATUS # The task status marker file. The content is Failed or Completed
│ └── TASK # The task description information
└── worker # Status of the task being run by the worker. After running, tasks are managed by the master
└── jobs
└── local_test
└── tasks
```

Note:

- For job running information, view **log/ossimport2.log**.
- For the task failure cause, view **master/jobs/${JobName}/failed_tasks/${TaskName}/audit.log**.
- For failed task files, view **master/jobs/${JobName}/failed_tasks/${TaskName}/error.list**.

FAQs

Refer to **FAQs**.

Download

Distributed deployment currently only supports Linux, and does not support Windows.

Download the tool for distributed deployment: ossimport-2.2.1.tar.gz.

Download the tool to a local directory and use the commandtar -zxvf ossimport-2.2.1.tar.gz -C $HOME/ossimport to unzip the files. The file structure after the unzipping is as follows:

```
ossimport
├── bin
│ ├── console.jar # The JAR package of the console module
│ ├── master.jar # The JAR package of the master module
│ ├── tracker.jar # The JAR package of the tracker module
│ └── worker.jar # The JAR package of the worker module
├── conf
│ ├── job.cfg # The template of the job configuration file
│ ├── sys.properties # Configuration file of the system running parameters
│ └── workers # Worker list
├── console.sh # The command line tool. Currently it only supports Linux
├── logs # Log directory
└── README.md # Description documentation. Read it carefully before use
```

Notice:

- OSS_IMPORT_HOME: The root directory of the OssImport. By default the directory is the **$HOME/ossimport** in the unzip command. You can also run the command export OSS_IMPORT_HOME=<dir> or modify the system configuration file **$HOME/.bashrc** to set the directory.
- OSS_IMPORT_WORK_DIR: The OssImport working directory. You can specify the directory through the configuration item **workingDir** in **conf/sys.properties**. The recommended values is **$HOME/ossimport/workdir**.
- Use absolute paths for OSS_IMPORT_HOME or OSS_IMPORT_WORK_DIR, such as /home/<user>/ossimport or /home/<user>/ossimport/workdir.

Configuration

The distributed version has three configuration files: conf/sys.properties, conf/local_job.cfg, and conf/workers. For descriptions of the configuration items, see the Introduction chapter.

conf/job.cfg: The configuration file template for the job in distributed mode. Modify the values according to the actual parameters prior to data migration.

conf/sys.properties: The configuration file for the system run parameters, such as the working directory and the worker running parameters

conf/workers: The worker list.

Notice:

- Confirm the parameters in **sys.properties** and **local_job.cfg** before submitting the job. The parameters in the job are not allowed to be changed after the job is submitted.
- Determine the worker list **workers** before starting the service. After the service is started, workers are not allowed to be added or deleted.

Running

Execute commands

In distributed deployment, the general steps for job execution are as follows:

1. Modify the job configuration file.
2. Deploy the service.
3. Clear jobs of the same name.
4. Submit the job.
5. Start the migration service.
6. View the job status.
7. Retry failed tasks.
8. Stop the migration job.

Detailed descriptions can be found below:

Deploy the service. Execute bash console.sh deploy in Linux.

Notice: Make sure the configuration files **conf/job.cfg** and **conf/workers** have been modified before deployment.

Clean jobs of the same name. If you ran a job of the same name before and want to execute the job again, clear the job with the same name first. If you have never run the job or you want to retry a failed job, do not run the clear command. Execute bash console.sh clean job_name in Linux.

Submit the data migration job. The OssImport does not support submitting jobs of the same

name. If there are jobs with the same name, use the clean command to clean the job with the same name first. To submit a job, you need to specify the job configuration file. The job' s configuration file template is at **conf/job.cfg**. We recommend that you modify the settings based on the template. Execute bash console.sh submit [job_cfg_file] in Linux and submit the job with the configuration file **job_cfg_file**. The **job_cfg_file** is an optional parameter. If not specified, the parameter is **$OSS_IMPORT_HOME/conf/job.cfg** by default. The **$OSS_IMPORT_HOME** is by default the directory where the **console.sh** file is located.

Start the service. Execute bash console.sh start in Linux.

View the job status. Execute bash console.sh stat in Linux.

Retry a failed task. Tasks may fail to run because of network issues or other causes. Only failed tasks will be retried. Execute bash console.sh retry [job_name] in Linux. The **job_name** is an optional parameter which specifies to retry tasks of the job named **job_name**. If the job_name parameter is not specified, failed tasks of all jobs will be retried.

Stop the service. Execute bash console.sh stop in Linux.

Notice:

- When the bash console.sh parameter has an error, the console.sh will automatically prompt the command format.
- We recommend that you use absolute paths for directories of the configuration file and submitted jobs.
- The configuration for jobs (that is, the configuration items in **job.cfg**) cannot be modified after submitted.

Common causes of job failure

A file in the source directory was modified during the upload process. This cause will be indicated by a SIZE_NOT_MATCH error in **log/audit.log**. In this case, the old file has been uploaded successfully, but the changes have not been synchronized to the OSS.

A source file was deleted during the upload process, leading to the download failure.

A source file name does not conform to naming rules of the OSS (file name cannot start with / or be empty), leading to the upload failure to the OSS.

The data source file fails to be downloaded.

The program exits unexpectedly and the job status is **Abort**. If this happens, contact after-sales technical support.

Job statuses and logs

After a job is submitted, the master splits the job into tasks, the workers execute the tasks and the tracker collects the task statuses. After a job is completed, the workdir directory contains the following:

```
workdir
├── bin
│ ├── console.jar # The JAR package of the console module
│ ├── master.jar # The JAR package of the master module
│ ├── tracker.jar # The JAR package of the tracker module
│ └── worker.jar # The JAR package of the worker module
├── conf
│ ├── job.cfg # The template of the job configuration file
│ ├── sys.properties # Configuration file of the system running parameters
│ └── workers # Worker list
├── logs
│ ├── import.log # Archive logs
│ ├── master.log # Master logs
│ ├── tracker.log # Tracker logs
│ └── worker.log # Worker logs
├── master
│ ├── jobqueue # Store jobs that have not been fully split
│ └── jobs # Store the job running status
│ └── xxtooss # Job name
│ ├── checkpoints # The checkpoint record that the master splits the job to tasks
│ │ └── 0
│ │ └── ED09636A6EA24A292460866AFDD7A89A.cpt
│ ├── dispatched # Tasks that have been assigned to the workers but haven't been fully run
│ │ └── 192.168.1.6
│ ├── failed_tasks # Tasks that failed to run
│ │ └── A41506C07BF1DF2A3EDB4CE31756B93F_1499348973217@192.168.1.6
│ │ ├── audit.log # The task running log. You can view the error causes in the log
│ │ ├── DONE # Mark of successful tasks. If the task fails, the mark is empty
│ │ ├── error.list # The task error list. You can view the error file list
│ │ ├── STATUS # The task status mark file. The content is Failed or Completed, indicating that the task failed or
succeeded
│ │ └── TASK # The task description information
│ ├── pending_tasks # Tasks that have not been assigned
│ └── succeed_tasks # Tasks that run successfully
│ └── A41506C07BF1DF2A3EDB4CE31756B93F_1499668462358@192.168.1.6
│ ├── audit.log # The task running log. You can view the error causes in the log
│ ├── DONE # Mark of successful tasks
│ ├── error.list # Task error list. If the task is successful, the list is empty
│ ├── STATUS # The task status mark file. The content is Failed or Completed, indicating that the task failed or
succeeded
│ └── TASK # The task description information
└── worker # Status of the task being run by the worker. After running, tasks are managed by the master
└── jobs
├── local_test2
```

```
| └── tasks
└── local_test_4
└── tasks
```

Note:

- For the job running status, view **logs/tracker.log**. For the worker running logs, view **logs/worker.log**. For the master running logs, view **logs/master.log**.-For the task failure cause, view **master/jobs/${JobName}/failed_tasks/${TaskName}/audit.log**.
- For failed task files, view **master/jobs/${JobName}/failed_tasks/${TaskName}/error.list**.

FAQs

Refer to FAQs.

This article mainly introduces the general application of OssImport and implementation of typical requirements.

Standalone and distributed modes

The OssImport has two deployment modes available: **standalone mode** and **distributed mode**. For small-scale data migration with the data size smaller than **30 TB**, the standalone mode is enough. Distributed mode is recommended for migration of a large data size.

Time-specific traffic limits

The worker offers a traffic limit feature. You can implement traffic limits through modifying the **workerMaxThroughput (KB/s)** item in the configuration file **sys.properties**. This configuration item does not take effect. You need to restart the service after the modification for the item to take effect. In distributed deployment mode, you need to modify the **sys.properties** in **$OSS_IMPORT_WORK_DIR/conf** for each worker and then restart the service.

You can implement timed modification to **sys.properties** through **crontab**, and then restart the service to implement time-specific traffic limits.

Add a worker

Determine the worker list before submitting the job. Currently OSS does not support adding workers dynamically.

Data validation without migration

The OssImport supports data validation without migration. The configuration item is jobType=audit

instead of import in the job configuration file **job.cfg** or **local_job.cfg**. Other configuration items are the same as data migration.

Seamlessly switch from a third-party storage service to the OSS

Following the steps below, you can switch from other storage services to the OSS seamlessly:

Full migration. At this point, the business is still running on the third-party storage service. Mark down the start time of the data migration T1. Note that the time should be in the **Unix timestamp** format, that is, the number of seconds since 00:00 UTC on January 1, 1970. You can get the value through the date +%s command.

Open the OSS image back-to-source feature. After the data migration is complete, set the **image back-to-source** feature for the **bucket** in the **OSS console**, and the back-to-source address is the third-party storage.

Switch reading/writing to the OSS. At this point, the data earlier than **T1** is read from the OSS, while the data later than **T1** is read from the third-party service using the image back-to-source, and new data is fully written to the OSS.

Incremental migration. The configuration item for an incremental migration jobs is importSince=T1 in the configuration file (**job.cfg** or **local_job.cfg**). The incremental migration is completed at **T2**.

Delete the third-party storage. After **T2**, all your business reads and writes occur on the OSS, and the third-party storage is only a copy of historical data. You can decide to keep it or remove it at your own discretion. The OssImport is responsible for data migration and validation and will not delete any data.

Notice: The incremental migration in Step 4 is not referring to the incremental mode of data migration.

Incremental mode of data migration

The incremental mode of data migration refers to the process of performing a full migration first after a data migration job is started and then performing incremental migration operations at set intervals automatically. The first data migration job is a full migration. The job is started immediately after it is submitted. The subsequent data migration jobs are initiated once every set interval. The incremental mode has two configuration items:

Check whether the isIncremental in **job.cfg** has enabled the incremental migration mode. If the value is true, it indicates that the incremental mode has been enabled; if the value is false, it indicates that the incremental mode has been disabled. The default setting is **false**.

The incrementalModeInterval setting in **job.cfg** specifies the synchronization interval in seconds for the incremental mode. The setting is used when isIncremental is set to **true**. The minimum value configurable is **900 seconds**. We do not recommend you configure it to a value smaller than **3,600 seconds** as that will waste a large number of requests and lead to additional system overhead.

Application scenarios:

- Data backup
- Data synchronization

Specify filtering conditions for object migration

Only objects that meet the specified filtering conditions are migrated. The OssImport supports specifying the prefix and last modified time:

The srcPrefix setting in **job.cfg** specifies the prefix of the objects to be migrated. It is empty by default. If the srcType is **local**, you need to fill in the local directory in full path and separate and end the input values with **/**, such as **c:/example/** or **/data/example/**. If the srcType is oss, qiniu, bos, ks3, youpai, or s3, you need to fill in the prefix of the objects to be synchronized, excluding the bucket name, such as **data/to/oss/**.

Notice: The srcPrefix of all objects should be set to empty.

The importSince option in **job.cfg** specifies the last modified time of the migration objects. It is an integer and expressed in seconds. If an object’ s Last Modified Time is at or before importSince, it will be migrated. If an object’ s Last Modified Time is after importSince, it will not be migrated. The importSince setting is in the **Unix timestamp** format, that is, the number of seconds since 00:00 UTC on January 1, 1970. You can get the value through the date +%s command. The default value is 0, indicating to migrate all the data. The incremental mode of data migration is only valid for the first full migration. The non-incremental mode is valid for the entire migration job.

Migrate local data to the OSS

Tools for migrating local data to the OSS:

If you want to migrate less than **30 TB** of local data files, or want to mount the storage service to a local file system, we recommend that you use **OssUtil**. The tool is easy and convenient to use. OssUtil supports incremental uploads at the **object level** and implements the feature through the -u/--update and --snapshot-path options. For detailed descriptions, run the ossutil help cp command to see details.

The distributed version of **OssImport** is recommended for migration of large scale data.

Notice: During incremental migration of local data, some operations of the file system won't modify the last modified time of objects, such as **cp** and **mv** in Windows, and **mv** and **rsync** with **-t** or **-a** options in Linux. Data changes from these operations will not be detected or synchronized to the OSS.

Data migration between OSS

When should I use OssImport:

If you want to add the **Cross-Region Replication** feature for data migration between OSS in different regions, you can configure the feature in the console.

If a region does not support **Cross-Region Replication** yet for policy or security reasons, you can use OssImport to migrate or back up data.

Data migration between different accounts and buckets within the same region.

We recommend Alibaba Cloud intranet for direct data migration within OSS, that is, using the ECS or OSS domain name with **internal**.

Charges for direct data migration within OSS:

If you use a domain name with **internal**, no traffic charges will be incurred, but you need to pay for the request and storage charges.

If you did not use a domain name with **internal**, traffic charges may be incurred.

Not recommended use cases:

Data migration between regions with the **Cross-Region Replication** service activated.

When you synchronize modifications to objects between OSS in incremental mode, the OssImport only supports synchronization of object modifications (put/append/multipart) and does not support synchronizing reading and deleting operations. The data synchronization is not guaranteed to be timely by a specific **SLA**. Exert caution when selecting this option. We recommend that you use Upload callback.

ECS and traffic

If you want to migrate data from the cloud (non-local) to the OSS and have insufficient bandwidth resources, we recommend that you buy Pay-As-You-Go ECS instances for the migration. ECS configuration:

- Select Pay-As-You-Go for the billing method.
- Select the corresponding region for the OSS.
- Select 100 MB for the bandwidth peak.
- Select either Linux or Windows as you need.
- Select the minimal configuration for all other items.

In migration job configuration, set the targetDomain to an intranet domain name containing **internal** . If the source end is also OSS, also set the srcDomain to an intranet domain name containing **internal** . This will save money for downloads from the OSS source domain name, and will only charge for OSS access.

Migrate HTTP data to OSS

Parameters to be configured for an HTTP data migration job:

In *job.cfg*, set srcType to srcType=http. It is case-sensitive.

In httpListFilePath of *job.cfg*, **use absolute paths** to specify the HTTP address list file, such as **c:/example/http.list** and **/root/example/http.list**. A full HTTP link is **127.0.0.1/aa/bb.jpg**. Different splitting methods may lead to different object paths on the OSS after the upload:

```
http://127.0.0.1/aa/   bb.jpg     # The first line http://127.0.0.1/     aa/bb.jpg   # The second line
```

The object name after the first line is imported to the OSS is **destPrefix + bb.jpg** and the object name of the second line is **destPrefix + aa/bb.jpg**. The *httpPrefixColumn* parameter specifies the domain name column. The first column applies by default, such as the aforementioned **127.0.0.1/aa/** or **127.0.0.1/**. The *relativePathColumn* specifies the object name in the OSS, such as the aforementioned **bb.jpg** or **aa/bb.jpg**. If there are multiple columns in the object, as follows:

```
```

```
http://127.0.0.1/aa/ bb/cc dd/ee ff.jpg
```

The configuration should be as below: *httpPrefixColumn=1* , *relativePathColumn=4*

- The destAccessKey, destSecretKey, destDomain, and destBucket configuration items among others in **job.cfg**.

Splitting parameters for HTTP data migration tasks:

taskObjectCountLimit: The maximum number of objects for each task. The default value is 10,000.

taskObjectSizeLimit: The maximum data size of each task. **This parameter is invalid for HTTP data migration**, because when the master is splitting tasks, if every HTTP object is the size of the object obtained from the source, each object will have one HTTP request overhead, which will negatively impact the task allocation efficiency, thereby compromising concurrent execution of tasks and migration efficiency.

Domain name: The first column in the object specified by httpListFilePath. Continuous jobs with the same domain name are split according to the taskObjectCountLimit parameter, and continuous jobs with different domain names are split into different tasks to make better reuse of connections. For example:

```
 http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/  import/test1.txt
http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/ import/test2.txt
http://mingdi-bj.oss-cn-beijing.aliyuncs.com/ import/test3.txt
http://mingdi-bj.oss-cn-beijing.aliyuncs.com/ import/test4.txt
```

When the taskObjectCountLimit value is greater than 2, the job will be split into two tasks, while

```
 http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/  import/test1.txt
http://mingdi-bj.oss-cn-beijing.aliyuncs.com/ import/test3.txt
http://mingdi-hz.oss-cn-hangzhou.aliyuncs.com/ import/test2.txt
http://mingdi-bj.oss-cn-beijing.aliyuncs.com/ import/test4.txt
```

will be split into four tasks. **That is why httpListFilePath specified HTTP address list objects are first sorted by domain name.**

Network traffic and parameter configuration

The configuration of the following parameters is related to network traffic:

- In *sys.properties*, the workerTaskThreadNum parameter indicates the number of jobs for concurrent execution by the *worker*. If the network quality is poor or the concurrency is high, there may be a large number of timeout errors. At this point, you are advised to reduce the concurrency, modify the configuration item and restart the service.
- In *sys.properties*, the workerMaxThroughput(KB/s) parameter indicates the traffic ceiling of the *worker*. If you want to limit the traffic, such as for traffic control on the source end, or out of network restrictions, the value of this parameter should be smaller than the maximum network traffic allowed for the machine and evaluated based on business requirements.
- In *job.cfg*, the taskObjectCountLimit parameter indicates the maximum number of objects of each *task*. The default value is 10,000. This parameter will influence the number of tasks. If the number of tasks is too small, the concurrent tasks may be less efficient.
- In *job.cfg*, the taskObjectSizeLimit indicates the maximum data size of each *task*. The default value is 1 GB. This parameter will influence the number of tasks. If the number of tasks is too small, the concurrent tasks may be less efficient.

Notice:

- We recommend that you determine the configuration file parameters before starting the migration.
- Modifications to parameters in **sys.properties** take effect after you restart the migration server.
- After the **job.cfg** job is submitted, the configuration parameters of the job cannot be changed.