

Alibaba Cloud Object Storage Service

Developer Guide

Issue: 20190617

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.
2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.
3. The content of this document may be changed due to product version upgrades, adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.
4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults" and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use

or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.

5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).
6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Generic conventions

Table -1: Style conventions

Style	Description	Example
	This warning information indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	 Danger: Resetting will result in the loss of user configuration data.
	This warning information indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	 Warning: Restarting will cause business interruption. About 10 minutes are required to restore business.
	This indicates warning information, supplementary instructions, and other content that the user must understand.	 Notice: Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructions, best practices, tips, and other content that is good to know for the user.	 Note: You can use Ctrl + A to select all files.
>	Multi-level menu cascade.	Settings > Network > Set network type
Bold	It is used for buttons, menus, page names, and other UI elements.	Click OK.
Courier font	It is used for commands.	Run the <code>cd / d C :/ windows</code> command to enter the Windows system folder.
<i>Italics</i>	It is used for parameters and variables.	<code>bae log list --instanceid Instance_ID</code>
[] or [a b]	It indicates that it is an optional value, and only one item can be selected.	<code>ipconfig [-all -t]</code>

Style	Description	Example
<code>{}</code> or <code>{a b}</code>	It indicates that it is a required value, and only one item can be selected.	<code>switch {stand slave}</code>

Contents

Legal disclaimer.....	I
Generic conventions.....	I
1 Usage instructions.....	1
2 Basic concepts.....	2
3 Endpoint.....	7
3.1 Endpoints.....	7
3.2 Regions and endpoints.....	10
4 Storage classes.....	16
4.1 Overview.....	16
4.2 Convert between storage classes.....	20
4.3 Create and use Archive buckets.....	23
5 Access OSS.....	31
5.1 Quick start.....	31
5.2 OSS-based app development.....	32
6 Compliant retention strategy.....	38
6.1 Introduction.....	38
6.2 Set a compliant retention strategy.....	40
6.3 FAQ.....	41
7 Disaster recovery.....	43
7.1 Redundant storage across zones.....	43
7.2 Cross-region replication.....	44
8 Buckets.....	47
8.1 Create a bucket.....	47
8.2 Set the ACL for a bucket.....	48
8.3 Obtain the region information of a bucket.....	50
8.4 View the bucket list.....	51
8.5 Enable the pay-by-requester mode.....	52
8.6 Bind a custom domain.....	54
8.7 Configure hotlinking protection.....	56
8.8 Set CORS rules.....	59
8.9 Delete a bucket.....	61
9 Upload files.....	62
9.1 Simple upload.....	62
9.2 Form upload.....	63
9.3 Multipart upload and resumable upload.....	67
9.4 Append upload.....	71
9.5 Authorized third-party upload.....	74
9.6 Upload callback.....	76

9.7 RTMP-based stream ingest.....	77
10 Download files.....	82
10.1 Simple download.....	82
10.2 Resumable download.....	83
10.3 Authorized third-party download.....	84
10.4 OSS Select.....	86
11 Manage files.....	87
11.1 Manage Object Meta.....	87
11.2 View the object list.....	88
11.3 Copy objects.....	92
11.4 Delete objects.....	94
11.5 Manage back-to-origin configurations.....	95
11.6 Use the SelectObject API.....	99
11.7 Object tagging.....	117
12 Signature.....	121
12.1 OSS request process.....	121
12.2 Add a signature to the header.....	127
12.3 Add a signature to a URL.....	134
13 Identity authentication.....	138
13.1 What is RAM and STS.....	138
13.2 RAM user.....	141
13.3 Access OSS with a temporary access token provided by STS.....	142
14 Access and control.....	152
14.1 Overview.....	152
14.2 Access control based on ACLs.....	152
14.3 Access control based on RAM Policy.....	156
14.3.1 Tutorial: Use RAM Policy to control access to buckets and folders.....	156
14.3.2 RAM policy.....	180
14.4 Bucket policy.....	191
14.5 Cross-account authorization.....	192
14.5.1 Overview.....	192
14.5.2 Tutorial:Authorize a RAM user under another Alibaba Cloud account by adding a bucket policy.....	192
15 Manage logs.....	195
15.1 Access logging.....	195
16 Data encryption.....	200
16.1 Server-side encryption.....	200
17 Static website hosting.....	205
17.1 Configure static website hosting.....	205
17.2 Tutorial: Host a static website using a custom domain name.....	207
18 OSS sandbox.....	216

19 Monitoring service.....	221
19.1 Monitoring service overview.....	221
19.2 Monitoring service user guide.....	223
19.3 Alert service.....	237
19.4 Metric item reference.....	242
19.5 Monitoring indicators reference.....	251
19.6 Service monitoring, diagnosis, and troubleshooting.....	262
20 Cloud data processing.....	281
21 Hide.....	283
21.1 Access control.....	283
21.1.1 Bucket permission separation.....	283
21.1.2 STS temporary access authorization.....	285
21.1.3 FAQ about subaccount settings.....	296

1 Usage instructions

If it is your first time using Alibaba Cloud OSS, see the [OSS Quick Start Guide](#) to quickly get started with OSS.

The following table lists the manuals and guides that help you fully utilize OSS:

Resource	Description
Developer Guide	Describes the core concepts, functions, and provides methods (through console, API, or SDK) of using these functions.
Best Practices	Describes the application scenarios and configuration practices of OSS.
SDK Reference	Describes the SDK development, related parameters, and code samples based on major languages.
API Reference	Describes the RESTful API operations supported by OSS and provides related examples.
Console User Guide	Describes all operations supported by the OSS console.
Image Processing Guide	Describes various functions provided by Image Processing, such as format conversion, cropping, scaling, rotation, watermarks, and style encapsulation.
OSS migration tool	Describes the official migration tool that helps you migrate data from your local or third-party storage service to OSS.

2 Basic concepts

Before you use OSS, we recommend that you have a basic understanding of the following concepts.

Bucket

A bucket is a container for objects stored in OSS. Every object is contained in a bucket. The data model structure of Alibaba Cloud OSS is flat instead of hierarchical.

- All objects (files) are directly related to their corresponding buckets. Therefore, OSS lacks the hierarchical structure of directories and subfolders as in a file system.
- A user can have multiple buckets.
- A bucket name must be globally unique within OSS and cannot be changed once a bucket is created.
- A bucket can contain an unlimited number of objects.

The naming conventions for buckets are as follows:

- The bucket names must contain only lower case letters, numbers, and hyphens (-).
- The bucket names must start and end with a lower-case letter or number.
- The bucket names must be at least 3 bytes and no more than 63 bytes in length.

Object

Objects, also known as files, are the fundamental entities stored in OSS. An object is composed of metadata, data, and key. The key is the unique object name in a bucket. Metadata defines the attributes of an object, such as the time last modified and the object size. You can also specify custom metadata of an object.

The lifecycle of an object starts when it is uploaded, and ends when it is deleted. During the lifecycle, the object content cannot be changed. If you want to modify an object, you must upload a new object with the same name as the existing one to replace it. Therefore, unlike the file system, OSS does not allow users to modify objects directly.

OSS provides the Append Upload function, which allows you to continually append data to the end of an object.

The naming conventions for objects are as follows:

- The object names must use UTF-8 encoding.
- The object names must be at least 1 byte and no more than 1023 bytes.
- The object names cannot start with a backslash (\) or a forward slash (/).



Note:

Object names are case sensitive. Unless otherwise stated, objects and files mentioned in OSS documents are collectively called objects.

Region

A region represents the physical location of an OSS data center. You can choose the region where OSS will store the buckets you create. You may choose a region to optimize latency, minimize costs, or address regulatory requirements. Generally, the closer the user is in proximity to a region, the faster the access speed is. For more information, see [OSS regions and endpoints](#).

Regions are configured at bucket level instead of object level. Therefore, all objects contained in a bucket are stored in the same region. A region is specified when a bucket is created, and cannot be changed once it is created.

Endpoint

An endpoint is the domain name used to access the OSS. OSS provides external services through HTTP RESTful APIs. Different regions use different endpoints. For the same region, access through an intranet or through the Internet also uses different endpoints. For example, regarding the Hangzhou region, its Internet endpoint is `oss-cn-hangzhou.aliyuncs.com`, and its intranet endpoint is `oss-cn-hangzhou-internal.aliyuncs.com`. For more information, see [OSS regions and endpoints](#).

AccessKey

An AccessKey (AK) is composed of an AccessKeyId and an AccessKeySecret. They work in pairs to perform access identity verification. OSS verifies the identity of a request sender by using the AccessKeyId/AccessKeySecret symmetric encryption method. The AccessKeyId is used to identify a user. The AccessKeySecret is used for the user to encrypt the signature and for OSS to verify the signature. The AccessKeySecret must be kept confidential. In OSS, AccessKeys are generated by the following three methods:

- The bucket owner applies for AccessKeys.

- The bucket owner uses RAM to authorize a third party to apply for AccessKeys.
- The bucket owner uses STS to authorize a third party to apply for AccessKeys.

For more information about AccessKeys, see [Access control](#).

Strong consistency

In OSS, object operations are atomic, which means operations are either successful or failed without an intermediate state. OSS will never write corrupted or partial data.

Object operations in OSS are strongly consistent. For example, once a user receives an upload (PUT) success response, the object can be read immediately, and the data has already been written in triplicate. Therefore, OSS provides strong consistency for read-after-write. The same is true for the delete operations. Once a user deletes an object, the object becomes nonexistent immediately.

Data redundancy mechanism

OSS uses a data redundancy storage mechanism to store redundant data of each object on multiple devices of different facilities in the same area, ensuring data reliability and availability in case of hardware failure.

- Object operations in OSS are strongly consistent. For example, once a user receives an upload or copy success response, the object can be read immediately, and the redundant data has already been written to multiple devices.
- To ensure complete data transmission, OSS checks whether an error occurs when packets are transmitted between the client and the server by calculating the checksum of the network traffic packets.
- The redundant storage mechanism of OSS can avoid data loss if two storage facilities are damaged at the same time.
 - After data is stored in OSS, OSS checks whether redundant data is lost. If yes, OSS recovers the lost redundant data to ensure data reliability and availability.
 - OSS periodically checks the integrity of data through verification to discover data damage caused by factors such as hardware failure. If data is partially damaged or lost, OSS reconstructs and repairs the damaged data by using redundant data.

Comparison between OSS and file systems

Comparison item	OSS	File system
Data model	OSS is a distributed object storage service that uses a key-value pair format.	The file system is a hierarchical tree structure of directories that contain files.
Data retrieval	<p>Objects are retrieved based on unique object names (keys).</p> <p>Although users can use names like test1/test.jpg , this does not indicate that the object test.jpg is saved in a directory named test1. For OSS, test1/test.jpg and a.jpg have no essential difference . Similar amounts of resources are consumed during access to objects of different names.</p>	Files are retrieved based on their locations in directories.
Advantage	OSS supports massive concurrent accesses , which means large volumes of unstructured data (such as images, videos, and documents) can be stored and retrieved without excessive use of resources.	Folder operations such as renaming, moving, and deleting directories are quite easy, because data does not need to be copied and replaced.

Comparison item	OSS	File system
Disadvantage	<p>The stored objects cannot be modified directly.</p> <p>If you want to modify an object, you must upload the new object of the same name to replace the existing one.</p>	<p>System performance depends on the capacity of a single device. The more files and directories that are created in the file system, the more resources are consumed, and the lengthier the user process becomes.</p>

As a result, mapping OSS to a file system is not a recommended practice. When you use OSS, we recommend that you make full use of its advantages, including its massive data processing capabilities to store massive volumes of unstructured data, such as images, videos, and documents.

The mapping between OSS concepts and file system concepts is as follows:

OSS	File system
Object	File
Bucket	Home directory
Region	NA
Endpoint	NA
AccessKey	NA
NA	Multilevel directory
GetService	Retrieving the list of home directories
GetBucket	Retrieving the list of files
PutObject	Writing a file
AppendObject	Appending data to an existing file
GetObject	Reading a file
DeleteObject	Deleting an object
NA	Modifying file content
CopyObject (same target and source)	Modifying file attributes
CopyObject	Copying a file
NA	Renaming a file

3 Endpoint

3.1 Endpoints

Composition rules for domain names

In the network requests for OSS, except those for the GetService API, the domain names are the third-level domain names with specified bucket names.

The domain name contains a bucket name and an endpoint in the format of BucketName.Endpoint. An endpoint is an access domain name. OSS provides external services through HTTP RESTful APIs. Different regions use different domain names. A region has an Internet endpoint and an intranet endpoint. For example, the Internet endpoint of the region China East 1 is oss-cn-hangzhou.aliyuncs.com, and the intranet endpoint of the region China East 1 is oss-cn-hangzhou-internal.aliyuncs.com. For more information, see [Regions and endpoints](#).

Access OSS through the Internet

You can always access OSS through the Internet. In the Internet, the inbound traffic (write) is free, and outbound traffic (read) is charged. For more information about outbound traffic charges, see [OSS Pricing](#).

You can access OSS through the Internet by using either of the following methods:

- Method 1: Use the URL to access OSS resource. The URL is constructed as follows:

```
< Schema >://< Bucket >.< Internet Endpoint >/< Object >, where :  
Schema is HTTP or HTTPS .  
Bucket is your OSS storage space .  
Endpoint is the access domain name for the region  
of a bucket . Enter the Internet endpoint here .
```

Object is a file uploaded to the OSS .

For example, in the region China East 1, the object named myfile/aaa.txt is stored in the bucket abc. The Internet access address of the object is:

```
abc . oss - cn - hangzhou . aliyuncs . com / myfile / aaa . txt
```

You can directly use the object URL in HTML as follows:

```
< img src =" https :// abc . oss - cn - hangzhou . aliyuncs . com / mypng / aaa . png " />
```

- Method 2: Configure the Internet access domain name through OSS SDKs.

You must set different endpoints when operating buckets of different regions.

For example, before configuring buckets in the region China East 1, you must set the endpoint during class instantiation.

```
String accessKeyId = "< key >";
String accessKeySecret = "< secret >";
String endpoint = " oss - cn - hangzhou . aliyuncs . com ";
OSSClient client = new OSSClient ( endpoint , accessKeyId , accessKeySecret );
```

Access OSS through an intranet

Intranet refers to the internal communication networks among Alibaba Cloud products. For example, you access OSS through ECS. In an intranet, the inbound and outbound traffic is free. If the ECS instance and the OSS bucket are in the same region , we recommend that you use an intranet to access OSS.

You can access OSS through an intranet by using either of the following methods:

- Method 1: Use the URL to access OSS resource. The URL is constructed as follows:

```
< Schema >://< Bucket >.< IntranetEndpoint >/< Object >, where :
Schema is HTTP or HTTPS .
Bucket is your OSS storage space .
Endpoint is the access domain name for the region of a bucket . Enter the intranet endpoint here .
```

Object is a file uploaded to the OSS .

For example, in the region China East 1, the object named myfile/aaa.txt is stored in the bucket abc. The intranet access address of the object is:

```
abc . oss - cn - hangzhou - internal . aliyuncs . com / myfile /
aaa . txt
```

- Method 2: Configure the intranet access domain name using OSS SDKs on ECS.

For example, set the intranet endpoint for the Java SDK on ECS as follows:

```
String  accessKeyI d = "< key >";
String  accessKeyS ecret = "< secret >";
String  endpoint = " oss - cn - hangzhou - internal . aliyuncs
. com ";
OSSClient client = new OSSClient ( endpoint , accessKeyI
d , accessKeyS ecret );
```



Note:

If you want to use an intranet to access OSS, the ECS instance and the OSS bucket must be in the same region.

For example, you have purchased ECS instances of China North 2 (Beijing), and you have two OSS buckets:

- One buckets is beijingres, and its region is China North 2 (Beijing). The intranet address `beijingres . oss - cn - beijing - internal . aliyuncs . com` can be used by ECS instances to access beijingres resources, and the traffic generated is free.
- The other bucket is qingdaores, and its region is China North 1 (Qingdao). The intranet address `qingdaores . oss - cn - qingdao - internal . aliyuncs . com` cannot be used by ECS instances to access qingdaores resources. The Internet address `qingdaores . oss - cn - qingdao . aliyuncs . com` must be used to access qingdaores resources, and the outbound traffic generated is charged.

3.2 Regions and endpoints

Regions indicate the regions where the data center of OSS is located. Endpoints indicate the domain names used by users to access OSS through Internet. This topic describes the mapping relationship between regions and endpoints.

Regions and OSS endpoints in classic networks

The following table describes the OSS Internet and intranet endpoints of each region in classic networks.

Region name	OSS region	Internet endpoint	Internet endpoint protocol	Intranet endpoint for ECS access	Intranet endpoint protocol
China East 1 (Hangzhou)	oss-cn-hangzhou	oss-cn-hangzhou.aliyuncs.com	HTTP and HTTPS	oss-cn-hangzhou-internal.aliyuncs.com	HTTP and HTTPS
China East 2 (Shanghai)	oss-cn-shanghai	oss-cn-shanghai.aliyuncs.com	HTTP and HTTPS	oss-cn-shanghai-internal.aliyuncs.com	HTTP and HTTPS
China North 1 (Qingdao)	oss-cn-qingdao	oss-cn-qingdao.aliyuncs.com	HTTP and HTTPS	oss-cn-qingdao-internal.aliyuncs.com	HTTP and HTTPS
China North 2 (Beijing)	oss-cn-beijing	oss-cn-beijing.aliyuncs.com	HTTP and HTTPS	oss-cn-beijing-internal.aliyuncs.com	HTTP and HTTPS
China North 3 (Zhangjiakou)	oss-cn-zhangjiakou	oss-cn-zhangjiakou.aliyuncs.com	HTTP and HTTPS	oss-cn-zhangjiakou-internal.aliyuncs.com	HTTP and HTTPS

Region name	OSS region	Internet endpoint	Internet endpoint protocol	Intranet endpoint for ECS access	Intranet endpoint protocol
China North 5 (Hohhot)	oss-cn-huhehaote	oss-cn-huhehaote.aliyuncs.com	HTTP and HTTPS	oss-cn-huhehaote-internal.aliyuncs.com	HTTP and HTTPS
China South 1 (Shenzhen)	oss-cn-shenzhen	oss-cn-shenzhen.aliyuncs.com	HTTP and HTTPS	oss-cn-shenzhen-internal.aliyuncs.com	HTTP and HTTPS
Hong Kong	oss-cn-hongkong	oss-cn-hongkong.aliyuncs.com	HTTP and HTTPS	oss-cn-hongkong-internal.aliyuncs.com	HTTP and HTTPS
US West 1 (Silicon Valley)	oss-us-west-1	oss-us-west-1.aliyuncs.com	HTTP and HTTPS	oss-us-west-1-internal.aliyuncs.com	HTTP and HTTPS
US East 1 (Virginia)	oss-us-east-1	oss-us-east-1.aliyuncs.com	HTTP and HTTPS	oss-us-east-1-internal.aliyuncs.com	HTTP and HTTPS
Asia Pacific SE 1 (Singapore)	oss-ap-southeast-1	oss-ap-southeast-1.aliyuncs.com	HTTP and HTTPS	oss-ap-southeast-1-internal.aliyuncs.com	HTTP and HTTPS
Asia Pacific SE 2 (Sydney)	oss-ap-southeast-2	oss-ap-southeast-2.aliyuncs.com	HTTP and HTTPS	oss-ap-southeast-2-internal.aliyuncs.com	HTTP and HTTPS

Region name	OSS region	Internet endpoint	Internet endpoint protocol	Intranet endpoint for ECS access	Intranet endpoint protocol
Asia Pacific SE 3 (Kuala Lumpur)	oss-ap-southeast-3	oss-ap-southeast-3.aliyuncs.com	HTTP and HTTPS	oss-ap-southeast-3-internal.aliyuncs.com	HTTP and HTTPS
Asia Pacific SE 5 (Jakarta)	oss-ap-southeast-5	oss-ap-southeast-5.aliyuncs.com	HTTP and HTTPS	oss-ap-southeast-5-internal.aliyuncs.com	HTTP and HTTPS
Asia Pacific NE 1 (Tokyo)	oss-ap-northeast-1	oss-ap-northeast-1.aliyuncs.com	HTTP and HTTPS	oss-ap-northeast-1-internal.aliyuncs.com	HTTP and HTTPS
Asia Pacific SOU 1 (Mumbai)	oss-ap-south-1	oss-ap-south-1.aliyuncs.com	HTTP and HTTPS	oss-ap-south-1-internal.aliyuncs.com	HTTP and HTTPS
EU Central 1 (Frankfurt)	oss-eu-central-1	oss-eu-central-1.aliyuncs.com	HTTP and HTTPS	oss-eu-central-1-internal.aliyuncs.com	HTTP and HTTPS
UK (London)	oss-eu-west-1	oss-eu-west-1.aliyuncs.com	HTTP and HTTPS	oss-eu-west-1-internal.aliyuncs.com	HTTP and HTTPS
Middle East 1 (Dubai)	oss-me-east-1	oss-me-east-1.aliyuncs.com	HTTP and HTTPS	oss-me-east-1-internal.aliyuncs.com	HTTP and HTTPS

**Note:**

- We recommend that you use third-level domain names that are in `bucket name + endpoint` format to share links or bind custom domain names (CNAME). For

example, the third-level domain name for a bucket named oss-sample in China East 2 (Shanghai) is oss-sample.oss-cn-shanghai.aliyuncs.com.

- When using SDKs, use `http ://` or `https :// + endpoint` as the initialization parameter. For example, we recommend that you use `http :// oss - cn - shanghai . aliyuncs . com` or `https :// oss - cn - shanghai . aliyuncs . com` as the initialization parameter of an endpoint in China East 2 (Shanghai). Do not use a third-level domain name, that is, `http :// bucket . oss - cn - shanghai . aliyuncs . com`, as the initialization parameter.
- By default, the Internet address "oss.aliyuncs.com" directs to the Internet endpoint of China East 1 (Hangzhou), and the intranet address "oss-internal.aliyuncs.com" directs to the intranet endpoint of China East 1 (Hangzhou).

Regions and OSS endpoints in VPC networks

ECS instances in VPC networks can use the following endpoints to access OSS.

Region name	OSS region	Endpoint in VPC networks	Protocol
China East 1 (Hangzhou)	oss-cn-hangzhou	oss-cn-hangzhou-internal.aliyuncs.com	HTTP and HTTPS
China East 2 (Shanghai)	oss-cn-shanghai	oss-cn-shanghai-internal.aliyuncs.com	HTTP and HTTPS
China North 1 (Qingdao)	oss-cn-qingdao	oss-cn-qingdao-internal.aliyuncs.com	HTTP and HTTPS
China North 2 (Beijing)	oss-cn-beijing	oss-cn-beijing-internal.aliyuncs.com	HTTP and HTTPS
China North 3 (Zhangjiakou)	oss-cn-zhangjiakou	oss-cn-zhangjiakou-internal.aliyuncs.com	HTTP and HTTPS
China North 5 (Hohhot)	oss-cn-huhehaote	oss-cn-huhehaote-internal.aliyuncs.com	HTTP and HTTPS

Region name	OSS region	Endpoint in VPC networks	Protocol
China South 1 (Shenzhen)	oss-cn-shenzhen	oss-cn-shenzhen-internal.aliyuncs.com	HTTP and HTTPS
Hong Kong	oss-cn-hongkong	oss-cn-hongkong-internal.aliyuncs.com	HTTP and HTTPS
US West 1 (Silicon Valley)	oss-us-west-1	oss-us-west-1-internal.aliyuncs.com	HTTP and HTTPS
US East 1 (Virginia)	oss-us-east-1	oss-us-east-1-internal.aliyuncs.com	HTTP and HTTPS
Asia Pacific SE 1 (Singapore)	oss-ap-southeast-1	oss-ap-southeast-1-internal.aliyuncs.com	HTTP and HTTPS
Asia Pacific SE 2 (Sydney)	oss-ap-southeast-2	oss-ap-southeast-2-internal.aliyuncs.com	HTTP and HTTPS
Asia Pacific SE 3 (Kuala Lumpur)	oss-ap-southeast-3	oss-ap-southeast-3-internal.aliyuncs.com	HTTP and HTTPS
Asia Pacific SE 5 (Jakarta)	oss-ap-southeast-5	oss-ap-southeast-5-internal.aliyuncs.com	HTTP and HTTPS
Asia Pacific NE 1 (Tokyo)	oss-ap-northeast-1	oss-ap-northeast-1-internal.aliyuncs.com	HTTP and HTTPS
Asia Pacific SOU 1 (Mumbai)	oss-ap-south-1	oss-ap-south-1-internal.aliyuncs.com	HTTP and HTTPS
EU Central 1 (Frankfurt)	oss-eu-central-1	oss-eu-central-1-internal.aliyuncs.com	HTTP and HTTPS
UK (London)	oss-eu-west-1	oss-eu-west-1-internal.aliyuncs.com	HTTP and HTTPS

Region name	OSS region	Endpoint in VPC networks	Protocol
Middle East 1 (Dubai)	oss-me-east-1	oss-me-east-1-internal.aliyuncs.com	HTTP and HTTPS

Usage of endpoints

- For the composition rules of OSS endpoints and the methods of accessing OSS through the Internet and intranet, see [Endpoints](#).
- If you are an ECS user and need to use an OSS intranet endpoint, see [How do ECS users use OSS intranet addresses?](#)

4 Storage classes

4.1 Overview

OSS provides three storage classes: Standard, infrequent access (IA), and Archive, to cover various data storage scenarios from hot data to cold data.

Standard

OSS Standard storage provides highly reliable, highly available, and high-performance object storage services that support frequent data access. The high-throughput and low-latency service response capability of OSS can effectively support access to hotspot data. Standard storage is the right choice for storing various images for social networking and sharing, and storing data for audio and video applications, large websites, and big data analysis.

The Standard storage class has the following features:

- Designed for 99.999999999% data reliability.
- Designed for 99.99% service availability.
- Delivers high-throughput and low-latency access performance.
- Supports HTTPS-based transmission.
- Supports Image Processing.

IA

OSS IA storage is suitable for storing long-lived, but less-frequently accessed data (an average of once or twice per month). With a storage unit price lower than Standard storage, IA storage is suitable for long-term backup of various mobile applications, smart device data, and enterprise data. It also supports real-time data access. Objects of the IA storage class have a minimum storage duration. OSS charges a fee if you delete objects that are stored for less than 30 days. Objects of the IA storage class have a minimum billable size. Objects smaller than 64 KB are charged as 64 KB. Data retrieval incurs charges.

The IA storage class has the following features:

- Designed for 99.999999999% data reliability.
- Designed for 99.99% service availability.

- Supports real-time access.
- Supports HTTPS-based transmission.
- Supports Image Processing.
- Requires a minimum storage duration and minimum billable size.

Archive

OSS Archive storage has the lowest price among the three storage classes. It is suitable for storing archival data for a long time (more than half a year recommended), such as medical images, scientific materials, and video footage. The data is infrequently accessed during the storage period. It takes about 1 minute to restore the data from the frozen status to the readable status. Objects of the Archive storage class have a minimum storage duration. OSS charges a fee if you delete objects that are stored for less than 60 days. Objects of the Archive storage class have a minimum billable size. Objects smaller than 64 KB are charged as 64 KB. Data retrieval incurs charges.

The Archive storage class has the following features:

- Designed for 99.999999999% data reliability.
- Designed for 99.99% service availability.
- Takes about 1 minute to restore the stored data from the frozen status to the readable status.
- Supports HTTPS-based transmission.
- Supports Image Processing, but data needs to be restored first.
- Requires a minimum storage duration and minimum billable size.

Comparison of storage classes

Item	Standard	IA	Archive
Data reliability	99.999999999%	99.999999999%	99.999999999%
Service availability	99.99%	99.99%	99.99% (restored data)
Minimum billable size of objects	Actual size of objects	64 KB	64 KB
Minimum storage duration	N/A	30 days	60 days

Item	Standard	IA	Archive
Data retrieval fee	No data retrieval fee	Charged by the size of retrieved data. Unit: GB	Charged by the size of restored data. Unit: GB
Data access	Real-time access with a latency in milliseconds	Real-time access with a latency in milliseconds	One minute after data is restored from the frozen status to the readable status
Image Processing	Supported	Supported	Supported after data is restored from the frozen status to the readable status

**Note:**

OSS charges a data retrieval fee based on the size of data read from the underlying distributed storage system. The data transmitted on the Internet is billed as part of the outbound traffic.

Supported APIs

API	Standard	IA	Archive
Bucket creation, deletion, and query			
PutBucket	Supported	Supported	Supported
GetBucket	Supported	Supported	Supported
DeleteBucket	Supported	Supported	Supported
Bucket ACL			
PutBucketAcl	Supported	Supported	Supported
GetBucketAcl	Supported	Supported	Supported
Bucket logging			
PutBucketLogging	Supported	Supported	Supported
GetBucketLogging	Supported	Supported	Supported
Bucket static website hosting			
PutBucketWebsite	Supported	Supported	Not supported
GetBucketWebsite	Supported	Supported	Not supported

API	Standard	IA	Archive
Bucket hotlinking protection			
PutBucketReferer	Supported	Supported	Supported
GetBucketReferer	Supported	Supported	Supported
Bucket lifecycle			
PutBucketLifecycle	Supported	Supported	Supported for data deletion only
GetBucketLifecycle	Supported	Supported	Supported
DeleteBucketLifecycle	Supported	Supported	Supported
Cross-region replication			
PutBucketReplication	Supported	Supported	Supported
Cross-origin resource sharing (CORS)			
PutBucketCors	Supported	Supported	Supported
GetBucketCors	Supported	Supported	Supported
DeleteBucketCors	Supported	Supported	Supported
Object operations			
PutObject	Supported	Supported	Supported
PutObjectACL	Supported	Supported	Supported
GetObject	Supported	Supported	Supported after data is restored from the frozen state to the readable state
GetObjectACL	Supported	Supported	Supported
GetObjectMeta	Supported	Supported	Supported
HeadObject	Supported	Supported	Supported
CopyObject	Supported	Supported	Supported
OptionObject	Supported	Supported	Supported
DeleteObject	Supported	Supported	Supported

API	Standard	IA	Archive
DeleteMultipleObjects	Supported	Supported	Supported
PostObject	Supported	Supported	Supported
PutSymlink	Supported	Supported	Supported
GetSymlink	Supported	Supported	Supported
RestoreObject	Not supported	Not supported	Supported
Multipart operations			
InitiateMultipartUpload	Supported	Supported	Supported
UploadPart	Supported	Supported	Supported
UploadPartCopy	Supported	Supported	Supported
CompleteMultipartUpload	Supported	Supported	Supported
AbortMultipartUpload	Supported	Supported	Supported
ListMultipartUpload	Supported	Supported	Supported
ListParts	Supported	Supported	Supported
Image Processing	Supported	Supported	Supported

4.2 Convert between storage classes

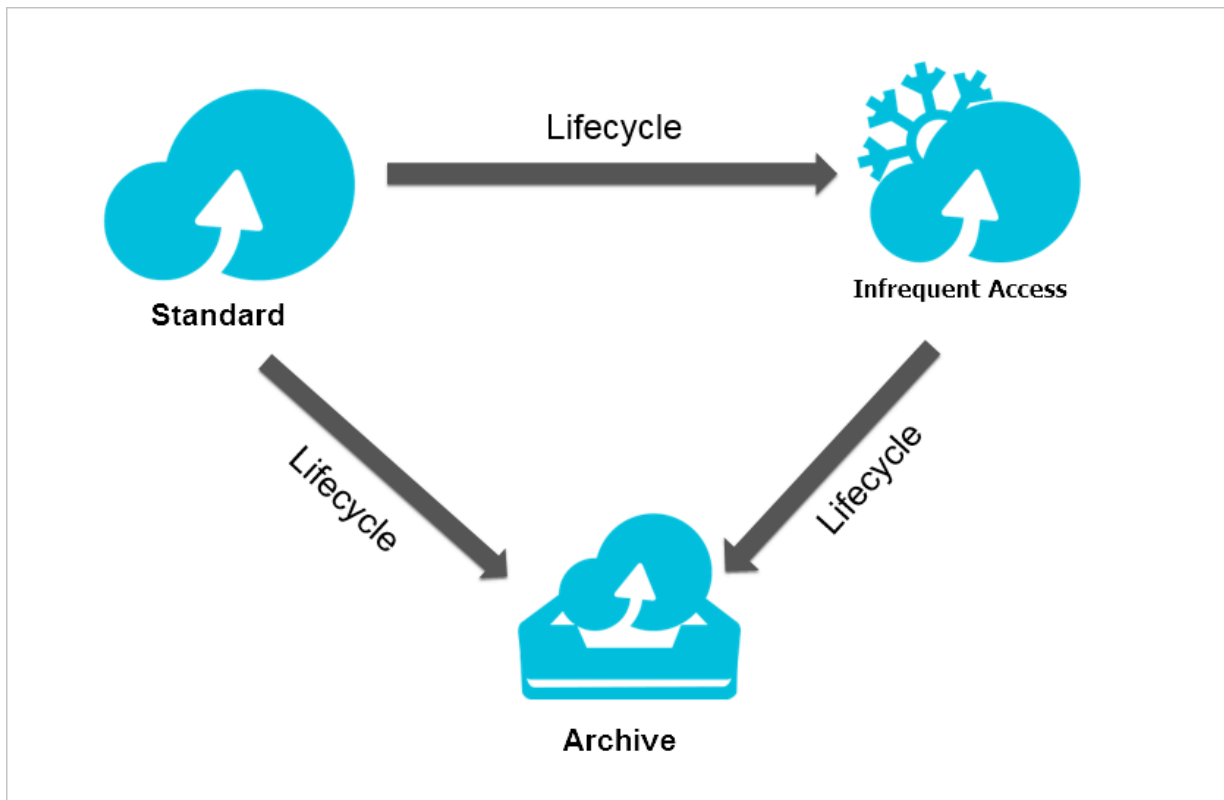
This topic describes how to convert the storage class of an object among Standard, infrequent access (IA), and Archive.

Lifecycle Object Transition

OSS supports three [storage classes](#): Standard, Infrequent Access, and Archive.

The Object Transition mechanism is now available in OSS Lifecycle Management function in all regions across China. The following storage classes are supported for automatic conversion:

- Standard -> Infrequent Access
- Standard -> Archive
- Infrequent Access -> Archive



Examples

You can configure lifecycle policies for objects with a given prefix in one bucket as follows:

- They are converted to Infrequent Access class after being stored for 30 days.
- They are converted to Archive class after being stored for 180 days.
- They are deleted automatically after being stored for 360 days.

You can complete the configuration of the preceding lifecycle policies in the console. For more information, see [Set lifecycle](#).

Create Lifecycle Rule

Status: **Enabled** Disabled

Policy: **Match by Prefix** Apply to Bucket

Prefix: video/

Delete File: **Expiration Period** Expiration Date

Not Enabled

Transition to IA After ☒ 30

Transition to Archive after ☒ 180

Delete All Objects After Specified Days ☒ 360

Delete Fragments: **Expiration Period** Expiration Date

Not Enabled

Delete Fragments After Specified Days 30

OK Cancel



Note:

If the following three parameters are configured:

Transition to IA After , Transition to Archive After , and Delete All Objects After Specified Days , then the number of days set for each parameter must meet the following criteria:

Days for converting to Infrequent Access < Days for converting to Archive < Specified days for deleting

Notes

After the Object type conversion, the storage cost is calculated based on the unit price of converted storage class.

Notes for Infrequent Access and Archive storage types:

- Minimum billable size:

Objects smaller than 128 KB are charged as 128 KB.

- Minimum storage period:

The stored data is required to be saved for at least 30 days. Charges will be incurred if you delete files that are stored for less than 30 days.

- Restore time of Archive type:

It takes one minute for Archive type Object to restore the data to a readable state. If real-time read is required in the business scenario, we recommend that you convert the file to the Infrequent Access storage class instead of Archive class. Otherwise, after converting the file to the Archive class, the data cannot be read in real time.

- Data access charges:

Both Infrequent Access and Archive classes are required to pay data access charges as a separate charge item to outbound traffic. If the average access frequency per Object is higher than once per month, you are not advised to convert the data to Infrequent Access or Archive class.

Storage classes conversion in other ways

For conversions from Archive type to Standard class or Infrequent Access class, or from Infrequent Access class to Standard class, you can read the Object and rewrite it to the Bucket of corresponding storage class. The default storage class of Object is determined by the Bucket.

For example, for the conversion of Infrequent Access Object in the Bucket of Standard type to Standard Object, you can read and rewrite the Object. Based on the type of the Bucket, the newly-written Object is of Standard storage class.

For the Object that has been converted to Archive class, you can only read it after performing Restore operation and restore it to a readable state.

For more information, see [Create and use Archive buckets](#).

4.3 Create and use Archive buckets

OSS provides three [storage classes](#). This topic describes how to create and use Archive buckets.

Create an Archive bucket

You can use the console, APIs or SDKs, or command-line tools to create an Archive bucket.

- Console

On the Create Bucket page of the OSS console, set Storage Class to Archive, as shown in the following figure.

Create Bucket ? Create a bucket ✕

Note: Storage Class and Region cannot be changed after the bucket is created.

Bucket Name 0/63

Region China (Beijing) ▾

Alibaba Cloud services in the same region can communicate with each other over an internal network. Select a region with caution because the region cannot be changed after the purchase.

Endpoint oss-cn-beijing.aliyuncs.com

Storage Class Standard IA Archive

Archive: long-term storage, very infrequent access, lowest storage price

[How to Choose a Suitable Storage Class](#)

Access Control List (ACL) Private Public Read Public Read/Write

Private: Authentication is required for users to read or write files.

OK Cancel

- APIs or SDKs

The following code uses the Java SDK as an example:

```
OSSClient ossClient = new OSSClient ( endpoint , accessKeyId , accessKeySecret );
CreateBucketRequest createBucketRequest = new CreateBucketRequest ( bucketName );
// Set the bucket ACL to public-read. The default ACL is private. createBucketRequest.setCannedACL ( CannedAccessControlList.PublicRead );
// Set the storage class to Archive. The default storage class is Standard.
createBucketRequest.setStorageClass ( StorageClass.Archive );
ossClient.createBucket ( createBucketRequest );
```

```
createBucketRequest.setStorageClass ( StorageClass.Archive );
```

); indicates that the storage class of the created bucket is Archive.

- Command-line tools

The following code uses `ossutil` as an example:

```
./ossutil mb oss://[Bucket name] --storage-class = Archive
```

Replace [Bucket name] with the name of your bucket to be created. Set `--storage-class` to `Archive` to create an Archive bucket.

Use an Archive bucket

- Upload data

You can use the `PutObject` and `MultipartUpload` APIs to upload data to an Archive bucket. The `AppendObject` API is not supported. Objects uploaded by using the `PutObject` and `MultipartUpload` APIs can be directly stored in Archive buckets.

- Download data

Different from objects stored in Standard and infrequent access (IA) buckets, objects stored in Archive buckets are not accessible in real time. To read an Archive

object, you must first submit a restore request and then wait for 1 minute until the data is readable.

The restore process of an Archive object is as follows:

1. The object is initially in the frozen state.
2. After you submit a restore request, the server starts to restore data, and the object enters the restoring state.
3. One minute later, the object is restored and enters the readable state.
4. The readable status lasts for one day by default, and can be prolonged to a maximum of seven days. After this period, the object returns to the frozen status .

You can restore an Archive object in any of the following ways:

- Console

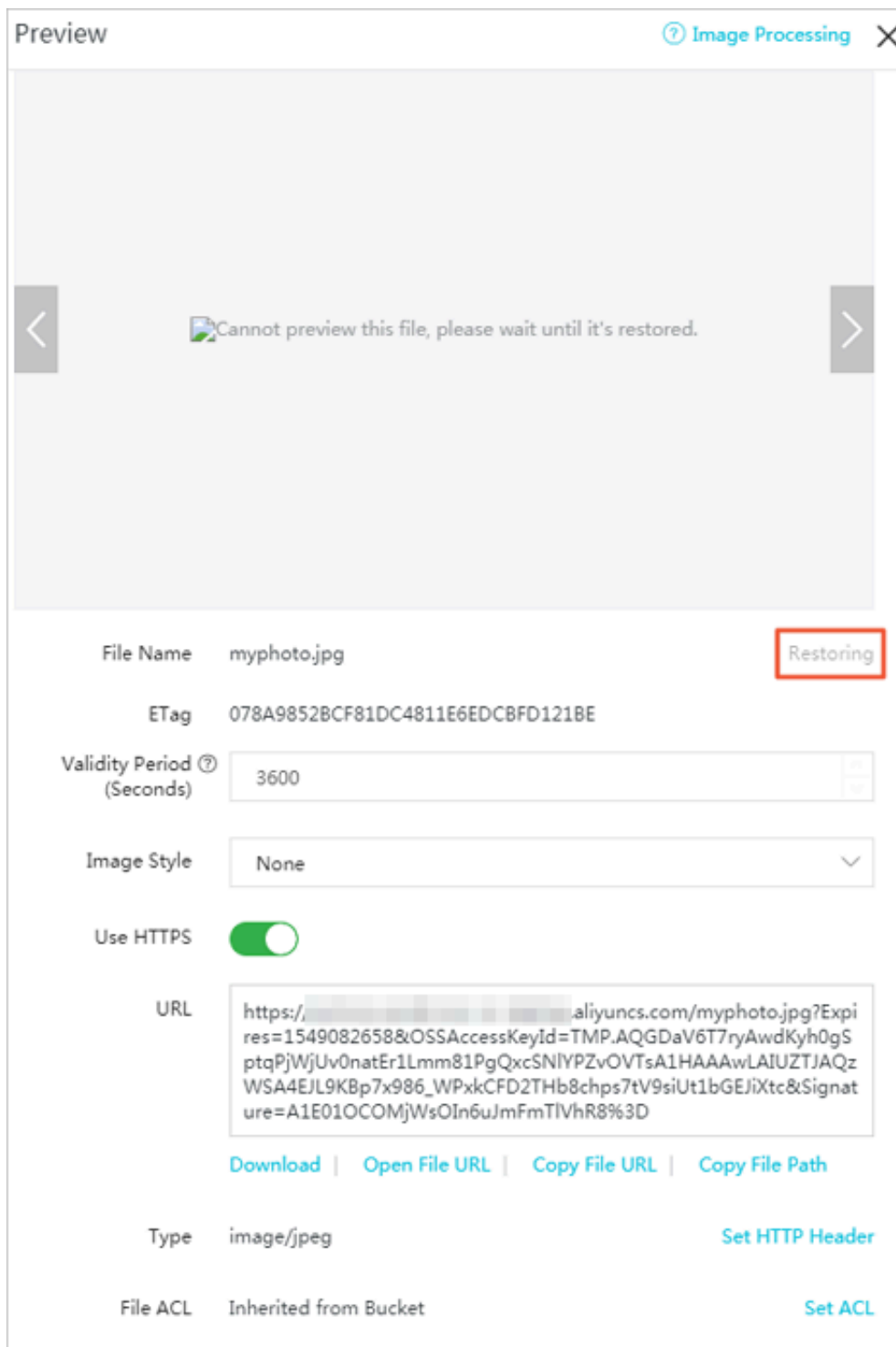
The screenshot displays the console interface for a file named 'myphoto.jpg'. The file's ETag is '078A9852BCF81DC'. The 'Validity Period (Seconds)' is set to '3600', and the 'Image Style' is 'None'. The 'Use HTTPS' toggle is turned on. The 'URL' field contains a long, complex string starting with 'https://'. Below the URL, there are four links: 'Download', 'Open File URL', 'Copy File URL', and 'Copy File Path'. The 'Type' is listed as 'image/jpeg'. A red box highlights the 'Restore' button in the top right corner. A modal dialog is open, showing the message: 'Restoring file myphoto.jpg. The restore operation will take about 1 minute. You can access the file after it is restored. Are you sure to restore it?'. The dialog has 'OK' and 'Cancel' buttons.

File Name	myphoto.jpg	Restore
ETag	078A9852BCF81DC	
Validity Period (Seconds)	3600	
Image Style	None	
Use HTTPS	<input checked="" type="checkbox"/>	
URL	https://[redacted]-beijing.aliyuncs.com/myphoto.jpg?Expires=1560761048&OSSAccessKeyId=TMP.AgFga3wz[redacted]BCOqX8n4UgsaCrraguzNCZu[redacted]CQkAAAwLAIUNi6KsIwRNCI3a_VbParYwIAXCrACFAaXWIR2ghb_Qpq1XKVys0KZBkPQ&Signature=WhevaE4[redacted]VMRwbjHLQ%3D	
	Download Open File URL Copy File URL Copy File Path	
Type	image/jpeg	

Click Preview in the Actions column corresponding to the destination object.

On the Preview page, click Restore. The restore operation takes about 1 minute.

During this process, the object is in the restoring state.



- APIs or SDKs

The following code uses the Java SDK as an example to show how to call the `restoreObject` method to restore an object:

```
ObjectMeta data objectMeta data = ossClient . getObjectM
etaddata ( bucketName , key );
// Check whether the storage class of the object is
Archive .
```

```
StorageClass storageClass = objectMetadata.getObjectStorageClass();
if (storageClass == StorageClass.Archive) {
    // Restore the object.
    ossClient.restoreObject(bucketName, key);
    // Wait until the restore operation is completed.
    do {
        Thread.sleep(1000);
        objectMetadata = ossClient.getObjectMetadata(
            bucketName, key);
    } while (!objectMetadata.isRestoreCompleted());
}
// Obtain the restored object.
OSSObject ossObject = ossClient.getObject(bucketName, key);
ossObject.getObjectContent().close();
```

- **Command-line tools**

The following code uses `ossutil` as an example:

```
./ossutil restore oss ://[ Bucket name ]/[ Object name ]
```

Replace `[Bucket name]` and `[Object name]` with the names of your bucket and object to be restored.

5 Access OSS

5.1 Quick start

This topic describes how to perform basic OSS operations, such as create a bucket, upload an object, and download an object.

1. Log on to the [OSS console](#).
2. Create a bucket.
3. Upload and download files.

For more information, see [Get started with Alibaba Cloud OSS](#).

Get familiar with OSS upload and download

Before you use OSS SDKs, we recommend that you have basic familiarity with the OSS upload and download methods.

OSS uses RESTful APIs to perform operations and all requests are standard HTTP requests.

OSS provides different upload methods to meet different requirements. You can:

- Use the Put Object method to upload a single file smaller than 5 GB to OSS. For more information, see [Simple upload](#).
- Use the Post Object method (HTTP form) to upload a file smaller than 5 GB to OSS from a browser. For more information, see [Form upload](#).
- Use the Multipart Upload method to upload a file larger than 5 GB. For more information, see [Multipart upload](#).
- Use the Append Object method to directly append content to the end of an object. This method is particularly well suited for video monitoring and live video broadcasting. For more information, see [Append object](#).

OSS also provides different download methods. For more information, see [Simple download](#) and [Multipart download](#).

General process of using OSS SDKs

1. Obtain the AccessKeyId and AccessKeySecret from the console.
2. Download the OSS SDKs in your preferred programming language from GitHub.

3. Perform uploads, downloads, and other operations.

For more information about how to use the OSS SDKs for different programming languages, see [OSS SDK Reference](#).

5.2 OSS-based app development

Development sequence diagram

Typical OSS-based app development involves the following four components:

- **OSS:** Provides functions such as upload, download, and upload callback.
- **Developer's mobile client** (mobile application or webpage application, called the client for short): Uses the service provided by the developer to access OSS.
- **Application server:** Interacts with the client. This server is used for the developer's service.
- **Alibaba Cloud STS:** Issues temporary credentials.

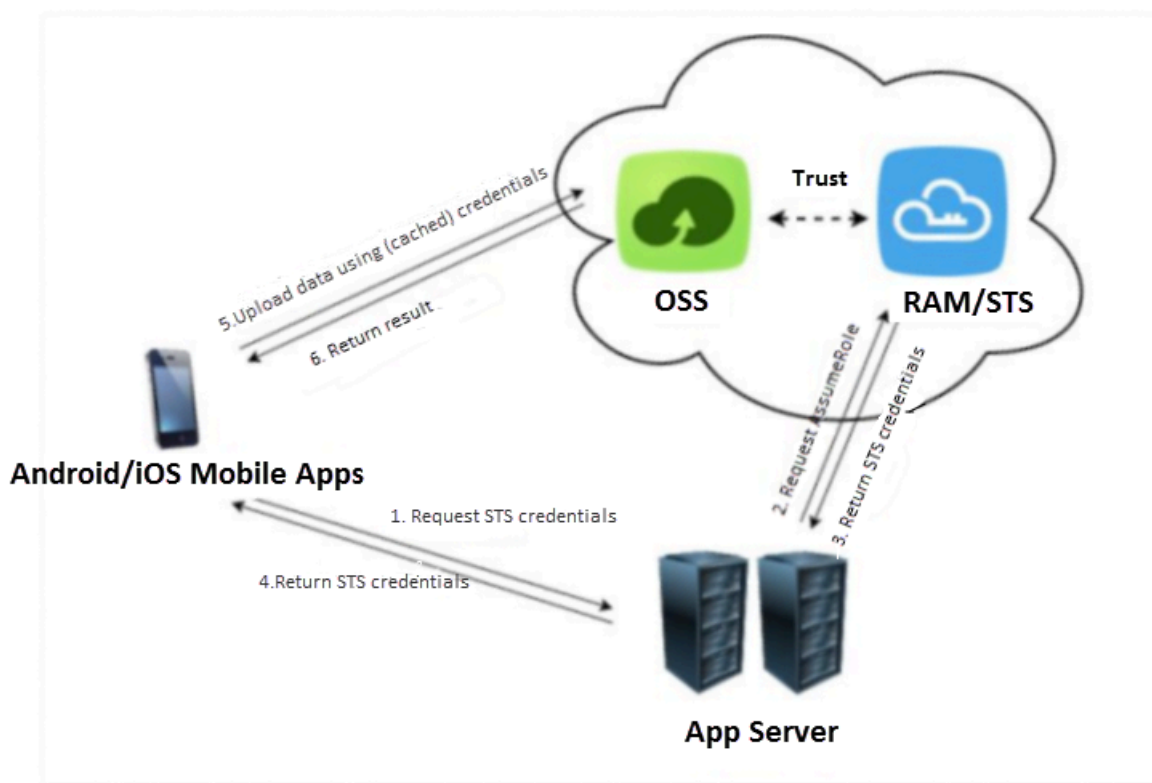
Best practices

- [Set up direct data transfer for mobile apps](#)
- [Set up data callback for mobile apps](#)
- [Permission control](#)

Service development process

- Data upload with temporary credential authorization

The following figure shows the process of data upload with temporary credential authorization:



The description of the process is as follows:

1. The client sends the application server the request of uploading data to OSS.
2. The application server sends a request to STS.
3. STS returns temporary credentials (STS AccessKey and token) to the application server.
4. The client obtains the authorization (STS AccessKey and token) and calls the mobile client SDK to upload data to OSS.
5. The client successfully uploads data to OSS. If callback is not set, the process is complete. If callback is set, OSS calls the relevant interface.

Note that:

- The client does not have to request authorization from the application server for each upload attempt. Each time the authorization is obtained, the client caches temporary credentials returned by STS until it expires.

- STS provides fine-grained access control for upload, which restricts client access permissions at the object level. This completely isolates the objects uploaded to OSS by different clients, and thus greatly enhances application security.

For more information, see [Authorized third-party upload](#).

- Data upload with signed URL or form authorization

The following figure shows the process of data upload with signed URL or form authorization:



The description of the process is as follows:

1. The client sends the application server the request of uploading data to OSS.
2. The application server returns credentials (signed URL or form) to the client.
3. The client obtains authorization (signed URL or form) and calls the mobile client SDK to upload data or uses form upload to directly upload data to OSS.
4. The client successfully uploads data to OSS. If callback is not set, the process is complete. If callback is set, OSS calls the relevant interface.

For more information, see [Authorized third-party upload](#).

- **Data download with temporary credential authorization**

The process of data download with temporary credential authorization is similar to that of data download with temporary credential authorization:

1. The client sends the application server the request of downloading data from OSS.
2. The application server sends a request to STS and obtains temporary credentials (STS AccessKey and token).
3. The application server returns the temporary credentials (STS AccessKey and token) to the client.
4. The client obtains the authorization (STS AccessKey and token) and calls the mobile client SDK to download data from OSS.
5. The client successfully downloads data from OSS.

Note that:

- For download, the client also caches temporary credentials to increase access speed.
- STS also provides fine-grained access control for download. The access control for both upload and download helps to isolate the OSS storage space for each mobile client.

- **Data download with signed URL authorization**

The process of signed URL authorization for download is similar to that of signed URL authorization for upload:

1. The client sends the application server the request of downloading data from OSS.
2. The application server returns the signed URL to the client.
3. The client obtains authorization (signed URL) and calls the mobile client SDK to download data from OSS.
4. The client successfully downloads data from OSS.



Note:

The client cannot store the developer's AccessKey. You can get only the URL signed by the application server or the temporary credentials issued with STS, that is, the AccessKey of the STS and token).

Best practices

- [What is RAM and STS](#)

Reference

- Android SDK: [Upload objects](#)
- iOS SDK: [Upload objects](#)

6 Compliant retention strategy

6.1 Introduction

Alibaba Cloud OSS fully supports the Write Once, Read Many (WORM) feature and allows users to store and use data in a manner that they cannot delete or modify the data, which conforms to the compliance requirements of Securities and Exchange Commission (SEC) and Financial Industry Regulation Authority (FINRA) of USA.



Note:

- As the only cloud service certificated by Cohasset Associates in China, Alibaba Cloud OSS is in compliance with the strict electronic archiving requirements of various regulations, such as EC Rule 17a-4(f), FINRA 4511, and CFTC 1.31. For more information, see [OSS Cohasset Assessment Report](#).
- The compliant retention strategy feature is now in the beta testing phase in the China South 1 (Shenzhen) region and will apply to all regions in the future.
- You can only set a compliant retention strategy for buckets in OSS.

OSS supports strong compliant retention strategies. You can set a time-based compliant retention strategy for your buckets. After the strategy is locked, all users can upload objects to the buckets or read objects in the bucket. However, before the retention period for the objects expires, the protected objects and the retention strategy cannot be deleted. You can delete an object protected by the retention strategy only when the retention period expires. The WORM feature supported by OSS applies to various industries, such as finance, insurance, medical, and securities. You can build a "cloud-based compliant storage space" based on OSS.



Note:

During the valid period of a compliant retention strategy, you can [set lifecycle rules](#) to convert the storage class of the objects in the bucket protected by the strategy, which reduces storage costs while ensuring the compliance.

Operation

You can set a compliant retention strategy in the OSS console. For more information, see [Set a compliant retention strategy](#).

Description

Currently, you can set only one time-based retention strategy with a retention period ranging from 1 day to 70 years.

Assume that you created a bucket named examplebucket on June 1, 2013, and then uploaded three objects named file1.txt, file2.txt, and file3.txt to the bucket at different times. After that, a compliant retention strategy was created for the bucket on July 1, 2014 with a protection period of 5 years. The following table describes the upload dates and expiration dates of the preceding three objects.

Object	Upload date	Expiration date
file1.txt	June 1, 2013	May 31, 2018
file2.txt	July 1, 2014	June 30, 2019
file3.txt	September 30, 2018	September 29, 2023

- Effective rules

After a time-based retention strategy is created for a bucket, it is in the InProgress state by default, and the state is valid for 24 hours. Within the validity period, resources that apply to the strategy in the bucket are protected.

- Within 24 hours after a compliant retention strategy is created for a bucket, the bucket owner and authorized users can delete the strategy if it is not locked. If the compliant retention strategy is locked, it cannot be deleted and the protection period of the strategy cannot be shorten. However, you can extend the protection period of the strategy.
- If a compliant retention strategy is not locked after it is created for 24 hours, it is not locked.
- If you try to delete or modify the data in a bucket protected by a compliant retention strategy, OSS API returns the 409 FileImmutable message.

- Deletion rules

- A time-based compliant retention strategy is a metadata property of a bucket. When a bucket is deleted, the compliant retention strategy and access strategies

set for the bucket are also deleted. Therefore, the owner of an empty bucket can delete the retention strategy set for the bucket by deleting the bucket.

- If a compliant retention strategy is not locked within 24 hours after it is created for a bucket, the bucket owner and authorized users can delete it.
- If a bucket stores objects which are in the protection period of a compliant retention strategy, you cannot delete neither the bucket nor the strategy set for the bucket.

6.2 Set a compliant retention strategy

A compliant retention strategy is used to specify the protection period of objects in a bucket. No one can modify or delete a protected object during the protection period.



Note:

- For more information about compliant retention strategies, see [Introduction](#).
- The compliant retention strategy feature is in the beta testing phase in the China South 1 (Shenzhen) region and will be supported in all regions.

Procedure

1. Log on to the [OSS console](#).
2. In the bucket list on the left, click the name of the bucket that you want to set a compliant retention strategy for.
3. Click the Basic Settings tab, locate the Retention Strategy area, and click Configure.
4. Click Create Strategy.
5. In the Create Strategy dialog box, set the Retention period for the compliant retention strategy. The value range of Retention period is 1 day to 70 years.
6. Click OK.



Note:

After a compliant retention strategy is created, it is in the IN_PROGRESS state. You can Lock or Delete a compliant retention strategy in this state.

7. Click Lock.
8. Confirm the compliant retention strategy and click OK.



Note:

- After this step, the strategy is in the LOCKED state. You can click Edit to extend the retention period.
- If you try to delete or modify the data in a bucket protected by a compliant retention strategy in the console, the `The file is locked and cannot be operated` error message is returned.

Calculate the retention time for an object

You can calculate the retention time for a protected object by adding the last update time of the object and the retention period specified in the compliant retention period. For example, a compliant retention strategy is set for a bucket, and the retention period specified in the strategy is 10 days. If the last update time of an object in the bucket is 2019-3-1 12:00, the object is protected until 2019-3-11 12:01. For more information about the compliant retention strategy, see [Introduction](#).

6.3 FAQ

This topic answers the questions that are frequently asked in the use the compliant retention strategy for Alibaba Cloud OSS resources.

What are the advantages of the compliant retention strategy?

The compliant retention strategy ensures that data is stored in a manner conforming to the compliance requirements of industry regulations. Data protected by a compliance retention strategy cannot be modified or deleted by any user within the protection period. However, if you use RAM policies or bucket policies to protect the data, it may still be modified or deleted.

When should I set a compliant retention strategy?

If you want to store important data that is not allow to modify or delete (such as medical records, technical documentations, and contracts) for a long period, you can store the data in a bucket and set a compliant retention strategy for the bucket to protect the data.

Can I set a compliant retention strategy for an object?

Currently, you can set a compliant retention strategy only for a bucket but not a directory or an object.

Can I modify the storage class of an object after setting a compliant retention strategy for the bucket that stores the object?

After setting a compliant retention strategy for a bucket, you can [set lifecycle rules](#) to convert the storage class of the objects in the protected bucket to save storage costs.

How do I delete a bucket for which a compliant retention strategy is set?

- You can directly delete the bucket if no objects are stored in it.
- If objects that are out of the protection period are stored in the bucket, a failure occurs when you try to delete the bucket. In this case, you can delete the objects in the bucket first and then delete the bucket.
- If objects that are in the protection period are stored in the bucket, you cannot delete the bucket.

Are my objects within the protection period of a compliant retention strategy kept when my account has overdue bills?

If your account has overdue bills, Alibaba Cloud applies data retention strategies to your data based on the terms and conditions in your contract.

Can I set a compliant retention strategy as an authorized RAM user?

APIs related to the compliant retention strategy are open and integrated into RAM policies. By using a RAM user authorized by RAM policies, you can create or delete a compliant retention strategy in the console or through the APIs and SDKs.

7 Disaster recovery

7.1 Redundant storage across zones

Data in OSS is separately stored in three zones within the same region. Users can access the data even if one of the three zones is unavailable. With this redundant storage mechanism, OSS achieves 99.999999999% data reliability and 99.99% data availability.

The redundant storage mechanism provides OSS with the disaster recovery capability in the data center level, that is, OSS can provide services with strong consistency even if a data center is not available because of network disconnection, power outage, or other disaster events. During failover, services are switched without interruption and data loss, ensuring that the failover process is not perceived by users. With the disaster recovery capability, OSS can meet the strict requirement that the Recovery Time Objective (RTO) and Recovery Point Objective (RPO) of key services must be 0.



Note:

Currently, redundant storage across zones is supported only in the China South 1 (Shenzhen) and China North 2 (Beijing) regions. If you want to use redundant storage across zones, [open a ticket](#) to apply for the trial of this function.

Supported storage class

Redundant storage across zones supports two storage classes: Standard and Infrequent Access (IA). The following table compares the two storage classes from different dimensions.

Index	Standard	IA
Data reliability	99.999999999%	99.999999999%
Designed service availability	99.99%	99.99%
Storage measurement unit	Actual object size	64 KB
Minimum storage period	No limit	30 days
Data retrieval fee	No fee is charged for data retrieval.	Charged based on the size of retrieved data (GB).

Index	Standard	IA
Data access	Real-time access with latency of milliseconds	Real-time access with latency of milliseconds
Image processing	Supported	Supported

Enable redundant storage across zones

You can enable redundant storage across zones for a bucket only when you create the bucket.

If you want to separately store data in an existing bucket in multiple zones for redundant storage, use data migration tools (such as ossimport and SDK) to copy the data to a bucket with the redundant storage across zones feature enabled.

7.2 Cross-region replication

This topic describes the application scenarios and limits of cross-region replication.

Bucket Cross-Region Replication enables automatic and asynchronous replication of objects across buckets in different OSS data centers (regions), which synchronizes operations (such as creating, updating, and deleting) performed on objects in the source bucket to the target bucket in a different region. This feature could be a boon to customers looking for cross-region disaster recovery for their buckets or data replication. Objects in the target bucket are precise copies of objects in the source bucket. They have the same object name, metadata, and content (such as creation time, owner, user-defined metadata, object ACL, and object content).



Notice:

- Currently, the cross-region replication feature is supported only between the regions in Mainland China and between the US West 1 (Silicon Valley) and US East 1 (Virginia) regions.
- The cross-region replication function is in the beta testing phase and is free currently. The function is charged after it is officially released.

Application scenarios

You may configure cross-region replication for your buckets for a variety of reasons, including:

- **Compliance requirements:** Although, by default, OSS stores multiple copies of each object on a physical disk, compliance requirements may dictate that you store a copy of the data at a further distance. Cross-region replication allows you to replicate data between distant OSS data centers to satisfy these compliance requirements.
- **Minimize latency:** Your customers are in two geographic locations. To minimize latency in accessing objects, you can maintain object copies in OSS data centers that are geographically closer to your users.
- **Data backup and disaster recovery:** You have high requirements on data security and availability, and you want to explicitly maintain copies of all written data in a second data center. In case one OSS data center is damaged by a catastrophic event like earthquake and tsunami, you can use backup data from the other one.
- **Data replication:** For business reasons, you may need to migrate data from one OSS data center OSS to another.
- **Operational reasons:** You have computing clusters in two different data centers that analyze the same set of objects. You may choose to maintain object copies in these regions.

Operation method

Operation method	Description
OSS console	Web application that is easy to use
Java SDK	Various and complete SDK demos in different languages

Usage

Cross-region replication supports synchronization of buckets with different names. If the two buckets are in different regions, you can use this feature to synchronize the data of the source bucket to the target one in real time. This feature now offers the following capabilities:

- **Real-time synchronization:** This provides the ability to monitor data additions, deletions, and modifications in real time and synchronize these changes to the target bucket. For files of 2 MB in size, data is synchronized in a matter of minutes to guarantee data consistency between the source and the target.
- **Historical data migration:** This provides the ability to synchronize historical data from a bucket to form two identical data copies.

- **Real-time display of synchronization progress:** This shows the point in time of the last synchronization for real-time data synchronization and the percentage of completion for historical data migration.
- **Easy configuration:** The OSS console provides an easy-to-use interface for configuration management.
- **Mutual synchronization:** You can achieve mutual synchronization between bucket A and bucket B as follows: Configure the synchronization from bucket A to bucket B first, and then configure the synchronization from bucket B to bucket A.

Restrictions

- For two buckets that are in synchronization, because you can operate on both buckets at the same time, copying an object from the source bucket may overwrite the object with the same name in the target bucket. Be cautious when using this feature.
- Because Bucket Replication uses an asynchronous copying method, it can take from minutes to hours to copy data to the target bucket, depending on the size of the objects being replicated.
- Cross-region synchronization only works when no synchronization to a third bucket is enabled for the two buckets to be synchronized. For example, if synchronization to Bucket B is enabled for Bucket A, you can no longer enable synchronization to Bucket C for Bucket A, unless you delete the former configuration first. Similarly, if synchronization to Bucket B is enabled for Bucket A, it is not allowed to enable synchronization from Bucket C to Bucket B.
- Synchronization is supported only between two buckets in different regions.
- Currently, the cross-region replication feature is supported only between the regions in Mainland China and between the US West 1 (Silicon Valley) and US East 1 (Virginia) regions.

8 Buckets

8.1 Create a bucket

Before uploading objects to OSS, you must use the PutBucket API of OSS to create a bucket to store objects. You can configure various attributes for a bucket, including the region, access permissions, and other metadata.



Note:

For more information about the PutBucket API, see [PutBucket](#).

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
ossbrowser	Graphical tool, which is easy to operate
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Android SDK	
iOS SDK	
Node.js SDK	
Ruby SDK	

Limits

- You can create a maximum of 30 buckets.

- The name of each bucket must be globally unique. To create a bucket, you must select a unique bucket name.
- Each bucket name must comply with the bucket naming rules.
- After a bucket is created, you cannot modify its name or region.

Access control

You can set the access control list (ACL) when creating a bucket, or modify the ACL for a created bucket. If you do not set the ACL, it is set to private by default. For more information, see [Set the ACL for a bucket](#).



Reference

- [Simple upload](#)
- [Simple download](#)
- [Delete a bucket](#)
- [Overview](#)

8.2 Set the ACL for a bucket

You can set the access control list (ACL) when creating a bucket, or modify the ACL for a created bucket based on your business needs. Only bucket owners can set or modify the ACL for buckets.

The following table describes the three types of ACLs for buckets.

ACL	Description	Access control
public-read-write	The public-read-write permission.	<p>Anyone (including anonymous users) can perform read and write operations on the objects in the bucket.</p> <div>  Warning: All users on the Internet can have access to the objects in the bucket and write data to the bucket. This may leak your bucket data and sharply increase your fees. If anyone maliciously writes illegal information, they may also infringe on your legitimate interests and rights. Therefore, we do recommend that you do not set your bucket ACL to public-read-write except for special needs. </div>
public-read	The public-read permission.	<p>Only the bucket owner can perform write operations on the objects in the bucket. Other users (including anonymous users) can perform only read operations on the objects in the bucket.</p> <div>  Warning: All users on the Internet can have access to the objects in the bucket. This may leak your bucket data and sharply increase your fees. Therefore, we recommend that you set your bucket ACL to public-read with caution. </div>
private	The private permission.	Only the bucket owner can perform read and write operations on the objects in the bucket. Other users have no access to the objects in the bucket.

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
ossbrowser	Graphical tool, which is easy to operate

Operating method	Description
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Node.js SDK	
Ruby SDK	

Reference

- [Overview](#)
- [Overview](#)

8.3 Obtain the region information of a bucket

You can use the GetBucketLocation API of OSS to obtain the location information about the region, that is, the data center, where a bucket is located.



Note:

For more information about the GetBucketLocation API, see [GetBucketLocation](#).

The returned Location field indicates the region where the bucket is located. For example, if a bucket is located in China (Hangzhou), the value of the returned Location field is `oss - cn - hangzhou`. For more information about regions, see [Regions and endpoints](#).

Operating methods

Operating method	Description
Console	Web application that displays the region of a bucket on the bucket overview page
ossbrowser	Graphical tool, which is easy to operate

Operating method	Description
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Node.js SDK	
Ruby SDK	

8.4 View the bucket list

After buckets are created, you can use the GetService API of OSS to obtain the bucket list.



Note:

For more information about the GetService API, see [GetService](#).

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
ossbrowser	Graphical tool, which is easy to operate
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	

Operating method	Description
Node.js SDK	
Ruby SDK	

Reference

- [Create a bucket](#)

8.5 Enable the pay-by-requester mode

After the pay-by-requester mode is enabled for a bucket in Alibaba Cloud OSS, the requester instead of the bucket owner pays the cost of the request and traffic. The bucket owner always pays the cost of storing data. You can enable this feature to share data without the need to pay for all requests and traffic by yourself.



Note:

The pay-by-requester mode is currently available only in China (Shenzhen).

Examples

- Share large datasets, such as ZIP code directories, reference data, geospatial information, or web crawling data. For example, a research institute provides a public dataset to share data with its customers and expects the customers to pay for requests and traffic. The configuration procedure is as follows:
 1. Enable the pay-by-requester mode for the bucket. For more information about the procedure, see [Enable the pay-by-requester mode for a bucket](#).
 2. Set a bucket policy to authorize the RAM users of your customers' Alibaba Cloud accounts to access your bucket. For more information about the configuration, see [Use bucket policies to authorize other users to access OSS resources](#).

- Deliver data to your customers or partners. For example, a company needs to deliver production data to its partners and expects the partners to pay for requests and traffic generated by data downloads.

The configuration procedure is as follows:

1. Enable the pay-by-requester mode for the bucket.
2. Set the bucket ACL to private.
3. Set a bucket policy to authorize the RAM users of your partners' Alibaba Cloud accounts to access your bucket. For more information about the configuration, see [Tutorial: Authorize a RAM user under another Alibaba Cloud account by adding a bucket policy](#).



Notice:

You must authorize the RAM users of your customers' or partners' Alibaba Cloud accounts to access your bucket, but not provide them with the AccessKey pair of your RAM user. If your customers or partners use your RAM user to access your bucket, you must pay for requests and traffic because the requester is still you.

Request methods

- Anonymous access is not allowed.

If you enable the pay-by-requester mode for a bucket, anonymous users are not allowed to access the bucket. Requesters must provide authentication information so that OSS can identify them and charge them, but not the bucket owner, for requests and traffic.

If requesters use the RAM role of an Alibaba Cloud account to request data, OSS charges this Alibaba Cloud account for such requests and traffic.

- Requesters must include the x-oss-request-payer field in the request header.

If you enable the pay-by-requester mode for a bucket, requesters must include the x-oss-request-payer:requester field in the request header for a POST, GET, or HEAD request. This field indicates that requesters understand that their requests and data downloads incur fees. Otherwise, requests cannot pass authentication.

The bucket owner does not need to include the x-oss-request-payer field in the request header when accessing their own bucket. In this case, the bucket owner is the requester who pays for such requests and traffic.

Fee analysis

In pay-by-requester mode, requesters pay for requests and traffic, and the bucket owner pays for data storage. However, in the following cases, requests fail (where HTTP status code 403 is returned), and OSS charges the bucket owner for such requests.

- The requester does not include the `x-oss-request-payer` field in the request header for a POST, GET, or HEAD request, or does not use this field as a parameter for a request through a RESTful API.
- The requester fails authentication.
- The requester is anonymous.

Reference

- Console: [Enable the pay-by-requester mode for a bucket](#)
- ossutil: [Access buckets in pay-by-requester mode](#)

8.6 Bind a custom domain

After you upload an object to a bucket in OSS, a URL is automatically generated for the object. You can use this URL to access the object in the bucket. To access an uploaded object by using a custom domain, you must bind the custom domain to the bucket where the object is stored and add a CNAME record that directs you to the Internet endpoint of the bucket.

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
PHP SDK	SDK demos in various languages
Node.js SDK	
Browser.js SDK	
Ruby SDK	

Common concepts

Before binding a custom domain, you must understand the following concepts:

- Custom domain : the domain that you buy from a DNS service provider.
- OSS endpoint or bucket endpoint : the domain that OSS assigns to your bucket. You can use this domain to access the resources in your bucket. To access an OSS bucket by using your user domain, you must bind the user domain to the OSS endpoint, that is, add a CNAME record to Alibaba Cloud Domain Name System (DNS).
- Alibaba Cloud CDN domain : The accelerating domain that Alibaba Cloud Content Delivery Network (CDN) assigns to your user domain. To use the CDN acceleration service to access the resources in your bucket, you must bind your user domain to a CDN accelerating domain, that is, add a CNAME record to Alibaba Cloud DNS.
- Auto CDN cache update : If you update an object in your bucket but cache duration of the object cached on the CDN node does not expire, users can only access the object that is not updated. In this case, you must manually update the object cache on the CDN node. To simplify operations, OSS provides the auto CDN cache update feature. After you enable this feature, all updates on the objects in your bucket are automatically synchronized to the CDN node. For more information, see [Enable auto CDN cache update](#).

Scenarios

For example, user A has a website with the domain `img . abc . com` , and the website contains an image with the URL of `http :// img . abc . com / logo . png` . For easier management, user A wants to redirect all requests for the image to OSS without modifying the code, that is, keep the URL of the image unchanged. In this case, user A can bind a custom domain. The process is as follows:

1. User A creates a bucket named `abc - img` in OSS and uploads the image to the bucket.
2. User A binds the custom domain `img . abc . com` to `abc - img` in the OSS console.
3. After `img . abc . com` is bound to `abc - img` , OSS maps the custom domain to the bucket.
4. User A adds a CNAME rule on the DNS server to map `img . abc . com` to `abc - img . oss - cn - hangzhou . aliyuncs . com` , which is the OSS endpoint of `abc - img` .

5. After receiving a request for `http://img.abc.com/logo.png`, OSS redirects the request to `abc-img` based on the mapping relationship between `img.abc.com` and `abc-img`. That is, users who access the image with the URL of `http://img.abc.com/logo.png` are redirected to the following URL: `http://abc-img.oss-cn-hangzhou.aliyuncs.com/logo.png`.

The following table compares the access processes before and after user A binds the custom domain.

	Before user A binds the custom domain	After user A binds the custom domain
Access process	<ol style="list-style-type: none"> 1. A user sends a request for <code>http://img.abc.com/logo.png</code>. 2. DNS resolves the IP address of user A's server from the request. 3. The user accesses the image <code>logo.png</code> on user A's server. 	<ol style="list-style-type: none"> 1. A user sends a request for <code>http://img.abc.com/logo.png</code>. 2. DNS resolves the OSS endpoint <code>abc-img.oss-cn-hangzhou.aliyuncs.com</code> from the request. 3. The user accesses the image <code>logo.png</code> in the OSS bucket <code>abc-img</code>.

Reference

- To set CDN acceleration, see [Bind a CDN accelerating domain](#).
- To access OSS resources from a static Webpage, see [Configure static website hosting](#) and [Tutorial: Use a custom domain to configure static website hosting](#).
- To access OSS resources through HTTPS, see [Certificate hosting](#).

8.7 Configure hotlinking protection

You can use the PutBucketReferer API of OSS to set the referer, so as to prevent unauthorized users from using your OSS data.



Note:

For more information about the PutBucketReferer API, see [PutBucketReferer](#).

To configure hotlinking protection, you need to set the following parameters:

- **Referer Whitelist:** the referer whitelist that specifies the domains that are allowed to access OSS resources.
- **Allow Empty Referer:** indicates whether the Referer field can be left empty in a request. If the Referer field cannot be left empty, users must include the Referer field in the HTTP or HTTPS request header so as to access OSS resources.

For example, you add `https://www.aliyun.com/` to the referer whitelist of a bucket named oss-example. Then, only users who set the Referer field to `https://www.aliyun.com/` in their requests can access the objects in this bucket.

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Node.js SDK	
Ruby SDK	

Detail analysis

- **Referer verification**
 - Hotlinking protection verification is required only when you access an object anonymously or by using a signed URL. Hotlinking protection verification is not required if the request header contains the `Authorization` field.
 - Hotlinking protection verification is required when the ACL of a bucket is private, public-read, or public-read-write.

- **Referer configuration**
 - A bucket supports multiple Referer parameters. When setting multiple Referer parameters, you can separate them with a line break in the console or with a comma (,) through the PutBucketReferer API.
 - You can use wildcards, such as an asterisk (*) or a question mark (?), to set the Referer parameter.
- **Referer effects**
 - If the referer whitelist is empty, OSS does not check whether the Referer field is left empty in requests. Otherwise, all requests are rejected.
 - If the referer whitelist is not empty and the Referer field cannot be left empty, OSS allows only requests whose Referer field values are in the referer whitelist, and rejects all other requests (including requests whose Referer fields are left empty).
 - If the referer whitelist is not empty and the Referer field can be left empty, OSS allows requests whose Referer fields are left empty and requests whose Referer field values are in the referer whitelist, and rejects all other requests.

Wildcards

- **Asterisk (*)**: used to replace zero or more characters. If you are looking for an object whose name is prefixed with AEW but have forgotten the remaining part, you can enter AEW* to search for all objects whose names start with AEW, such as *AEWT . txt* , *AEWU . EXE* , and *AEWI . dll* . To narrow down the search scope, you can enter AEW*.txt to search for all .txt objects whose names start with AEW, such as *AEWIP . txt* and *AEWDF . txt* .
- **Question mark (?)**: used to replace one character. If you enter love?, all objects whose names start with love and end with one character are displayed, such as lovey and lovei. To narrow down the search scope, you can enter love?.doc to search for all .doc objects whose names start with love and end with one character, such as *lovey . doc* and *loveh . doc* .

Reference

- For more information about the hotlinking protection principles, see [Anti-leech](#).
- For more information about FAQs for hotlinking protection, see [OSS hotlinking protection \(referer\) errors and troubleshooting](#).

8.8 Set CORS rules

Cross-origin resource sharing (CORS) is a standard cross-origin solution provided by HTML5. OSS uses the CORS standard for cross-origin access. You can use the PutBucketcors API of OSS to set CORS rules.



Note:

- For more information about the PutBucketcors API, see [CORS](#).
- For more information about CORS rules, see [W3C CORS specification](#).

Browsers that support JavaScript use the same-origin policy to ensure security. When website A uses the JavaScript code on its Webpage to access website B of another origin, the browser rejects the request. In this case, you can set CORS rules to allow cross-origin requests.

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	

Scenarios

Cross-origin access is commonly used in actual scenarios.

For example, a user uses OSS at the back end of the website [www.a.com](#). On the Webpage, objects can be uploaded by using JavaScript. However, requests on the upload Webpage can only be sent to [www.a.com](#). The browser rejects all requests sent to other websites. As a result, data uploaded by the user must be transferred through [www.a.com](#). If cross-origin access is configured, data can be directly uploaded to OSS, without the need to be transferred through [www.a.com](#).

CORS implementation

CORS is implemented as follows:

1. If CORS is enabled, an HTTP request contains the Origin field in its header to specify the origin. In the previous example, the Origin field is set to `www.a.com`.
2. After receiving the request, the server determines whether to allow the request from the origin based on CORS rules. If the request is allowed, the server includes the Access-Control-Allow-Origin field whose value is `www.a.com` in the response header, indicating that this cross-origin access is allowed. If the server allows all cross-origin requests, it includes the Access-Control-Allow-Origin field whose value is an asterisk (*) in the response header.
3. The browser checks whether the Access-Control-Allow-Origin field is returned in the response header to determine whether the cross-origin request is allowed. If this field is not returned, the browser blocks the request. If the request is not a simple one, the browser first sends an OPTIONS request to obtain the CORS configuration of the server. If the server does not allow subsequent CORS operations, the browser blocks subsequent not-so-simple requests.

OSS allows you to set CORS rules so as to determine whether to allow or reject cross-origin requests as needed. You can set CORS rules at the bucket level. For more information, see [PutBucketcors](#).

Detail analysis

- Browsers automatically include CORS-related header fields. You do not need to perform other operations. CORS operations are applicable only to browsers.
- Whether a CORS request is allowed is completely independent of OSS authentication. OSS uses CORS rules only to determine whether CORS-related header fields are included. Browsers determine whether to block such a request.
- Before sending cross-origin requests, you must ensure that the cache feature is disabled for browsers. For example, two Webpages in the same browser, one originated from `www.a.com` and the other from `www.b.com`, send a request for the same cross-origin resource simultaneously. If the server receives the request from `www.a.com` first, it returns the resource with the Access-Control-Allow-Origin field whose value is `www.a.com` in the response header. When the server receives the request from `www.b.com` later, the browser returns the cached resource with the Access-Control-Allow-Origin field whose value is `www.a.com` in the response

header. Because the response header field value does not match CORS rules for the request from www.b.com, this request fails.

FAQ

To troubleshoot common errors, see [OSS CORS errors and troubleshooting](#).

8.9 Delete a bucket

You can use the DeleteBucket API of OSS to delete a bucket that you create.



Note:

For more information about the DeleteBucket API, see [DeleteBucket](#).

Before deleting a bucket, you must delete all objects or object parts generated by incomplete multipart upload in the bucket. To delete all objects in a bucket, we recommend that you [manage object lifecycle](#).

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
ossbrowser	Graphical tool, which is easy to operate
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Android SDK	
iOS SDK	
Node.js SDK	
Ruby SDK	

9 Upload files

9.1 Simple upload

By using simple upload, you can use the PutObject API of OSS to upload single objects. Simple upload is applicable to scenarios where you can send an HTTP request to complete an upload, for example, to upload an object that is smaller than 5 GB.



Note:

- For more information about the PutObject API, see [PutObject](#).
- To upload an object that is larger than 5 GB, you can use [Resumable upload](#).

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
ossbrowser	Graphical tool, which is easy to operate
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Android SDK	
iOS SDK	
Node.js SDK	
Browser.js SDK	
Ruby SDK	

Upload limits

- **Size:** The maximum size of an object is 5 GB in this mode.
- **Naming rules:**
 - Object names must be UTF-8 encoded.
 - Object names must be one byte to 1,023 bytes in length.
 - Object names cannot start with a forward slash (/) or a backslash (\).

Object Meta setting

When using simple upload, you can set Object Meta to describe an object, for example, Content-Type and other standard HTTP header fields. You can also set user-defined information. For more information, see [Manage Object Meta](#).

Upload security and authorization

To prevent unauthorized third-party users from uploading data to your bucket, OSS provides bucket- and object-level access control. For more information, see [Access control](#).

To authorize third-party users to upload objects, OSS also provides account authorization. For more information, see [Authorized third-party upload](#).

Subsequent operations

- After uploading objects to OSS, you can use [upload callback](#) to submit a callback request to the specified application server and perform subsequent operations.
- After uploading images, you can use [Image Processing](#).
- After uploading audio or video objects, you can use [ApsaraVideo for Media Processing](#).

9.2 Form upload

By using form upload, you can use the PostObject API of OSS to upload objects with a maximum size of 5 GB.



Note:

For more information about the PostObject API, see [PostObject](#).

Scenarios

This method can be used on HTML Webpages to upload objects. A typical scenario is Web applications. For example, the following table compares the form upload process with other upload processes on a job-search website.

	Other upload methods	Form upload
Access process	<ol style="list-style-type: none">1. A website user sends a request to upload a resume.2. The website server responds with a resume upload page.3. The resume is uploaded to the website server.4. The website server uploads the resume to OSS.	<ol style="list-style-type: none">1. A website user sends a request to upload a resume.2. The website server responds with a resume upload page.3. The resume is uploaded to OSS.

- The form upload process is easier, because data is directly uploaded to OSS without being forwarded by the website server.
- In other upload processes, objects are uploaded to the website server first. When a large number of objects are uploaded, the website server must be scaled out. In the form upload process, objects are directly uploaded from the client to OSS. When a large number of objects are uploaded, OSS bears the upload pressure and ensures the service quality.

SDK demo

For more information, see [Java SDK](#).

Upload limits

- **Size:** The maximum size of an object is 5 GB in this mode.
- **Naming rules:**
 - Object names must be UTF-8 encoded.
 - Object names must be one byte to 1,023 bytes in length.
 - Object names cannot start with a forward slash (/) or a backslash (\).

Process analysis

1. Create a POST policy.

The policy form field of a POST request is used to verify the validity of the request. For example, you can set a policy to specify the size and name of the object to be

uploaded, and the URL that the client jumps to and the HTTP status code that the client receives after a successful upload. For more information, see [POST policy](#).

For example, in the following policy, you set the expiration time before which website users can upload data to 2115-01-27T10:56:19Z and the maximum size of the object to be uploaded to 104,857,600 bytes. (To ensure successful testing, a long expiration period is set, which is not recommended in actual use.)

```
This example uses the Python code . The policy is
a JSON - formatted string .
policy = "{\" expiration \":\" 2115 - 01 - 27T10 : 56 : 19Z \",\"
conditions \":[[\" content - length - range \", 0 , 104857600
]]}\"
```

2. Use Base64 to encode the policy string.
3. Use the AccessKey Secret of OSS to add a signature to the Base64-encoded policy.
4. Create an HTML page for upload.
5. Open the HTML page and select the object to be uploaded.

The following example shows the complete Python sample code.

```
# coding = utf8
import md5
import hashlib
import base64
import hmac
from optparse import OptionParser
er

def convert_base64 ( input ):
    return base64 . b64encode ( input )
def get_signature ( key , policy ):
    return base64 . b64encode ( hmac . new ( key , policy ,
hashlib . sha1 ). digest () )
def get_form ( bucket , endpoint , access_key _id , access_key
_secret , out ):
    # 1 Create a POST policy .
    policy = "{\" expiration \":\" 2115 - 01 - 27T10 : 56 : 19Z \",\"
conditions \":[[\" content - length - range \", 0 , 1048576 ]]]\"
    print ( " policy : % s " % policy )
    # 2 Use Base64 to encode the policy string .
    base64policy = convert_base64 ( policy )
    print ( " base64_encoded_policy : % s " % base64policy )
    # 3 Use the AccessKey Secret of OSS to add a
signature to the encoded policy .
    signature = get_signature ( access_key _secret ,
base64policy )
    # 4 Create an HTML page for upload .
    form = ''
    < html >
        < meta http - equiv = content - type content = " text /
html ; charset = UTF - 8 ">
        < head >< title > OSS form upload ( through the
PostObject API )< / title >< / head >
        < body >
            < form action = " http ://% s .% s " method = " post "
enctype = " multipart / form - data ">
```

```

        < input    type =" text "   name =" OSSAccessKeyID "
value ="% s ">
        < input    type =" text "   name =" policy "   value ="%
s ">
        < input    type =" text "   name =" Signature "   value
=" % s ">
        < input    type =" text "   name =" key "   value ="
upload /${ filename }">
        < input    type =" text "   name =" success_ac
tion_redirect "   value =" http :// oss . aliyun . com ">
        < input    type =" text "   name =" success_ac
tion_status "   value =" 201 ">
        < input    name =" file "   type =" file "   id =" file
">
        < input    name =" submit "   value =" Upload "   type ="
submit ">
    </ form >
</ body >
</ html >
''' % ( bucket , endpoint , access_key_id , base64policy ,
signature )
f = open ( out , " wb " )
f . write ( form )
f . close ()
print ( " form is saved into % s " % out )
if __name__ == '__main__':
    parser = OptionParser ()
    parser . add_option ( "", "-- bucket " , dest =" bucket " , help
=" specify " )
    parser . add_option ( "", "-- endpoint " , dest =" endpoint " ,
help =" specify " )
    parser . add_option ( "", "-- id " , dest =" id " , help ="
access_key_id " )
    parser . add_option ( "", "-- key " , dest =" key " , help ="
access_key_secret " )
    parser . add_option ( "", "-- out " , dest =" out " , help =" out
put form " )
    ( opts , args ) = parser . parse_args ()
    if opts . bucket and opts . endpoint and opts . id
and opts . key and opts . out :
        get_form ( opts . bucket , opts . endpoint , opts . id ,
opts . key , opts . out )
    else :
        print " python % s -- bucket = your - bucket -- endpoint
= oss - cn - hangzhou . aliyuncs . com -- id = your - access - key -
id -- key = your - access - key - secret -- out = out - put - form
- name " % __file__

```

Save this sample code as `post_object.py` and run it by using `python post_object.py`.

```

Usage :
python post_object . py -- bucket = Your bucket name --
endpoint = Bucket endpoint -- id = Your AccessKey ID -- key =
Your AccessKey Secret -- out = Output object name
Example :
python post_object . py -- bucket = oss - sample -- endpoint
= oss - cn - hangzhou . aliyuncs . com -- id = tphpxp -- key =
ZQNJzf4QJR krH4 -- out = post . html

```



Note:

- In the created form, `success_action_redirect` value = `http://oss.aliyun.com` indicates the Webpage that appears after the upload succeeds. You can replace it with your own Webpage.
- In the created form, `success_action_status` value = `201` indicates that HTTP status code 201 is returned after the upload succeeds. You can change it to another HTTP status code.
- If the generated HTML page is `post.html`, open `post.html` and select the object to be uploaded. In this example, OSS product page (<http://oss.aliyun.com>) appears after the upload succeeds.

Upload security and authorization

To prevent unauthorized third-party users from uploading data to your bucket, OSS provides bucket- and object-level access control. For more information, see [Access control](#).

To authorize third-party users to upload objects, OSS also provides account authorization. For more information, see [Authorized third-party upload](#).

9.3 Multipart upload and resumable upload

By using multipart upload and resumable upload provided by Alibaba Cloud OSS, you can split an object into multiple data blocks (parts) and upload them separately. After uploading all the object parts, you can call an API to combine them into an object.

Scenarios

When you use simple upload (through the `PutObject` API) to upload a large object to OSS, the upload may fail due to a network error. In the second upload attempt, you must upload the object from the beginning. In this case, you can use multipart upload to resume upload from the last uploaded part.

Compared with other upload methods, multipart upload is applicable to the following scenarios:

- Poor network connectivity: If the upload of one part fails on a mobile phone, you can re-upload only the failed part but not all parts.
- Resumable upload required: An upload in progress can be paused and resumed at any time.

- **Upload acceleration:** When the object to be uploaded to OSS is large, multiple parts can be uploaded concurrently to speed up the process.
- **Streaming upload:** Objects of unknown sizes can be uploaded at any time. This scenario is common in industry applications such as video surveillance.

Multipart upload

Operating method	Description
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	

Resumable upload

If the system crashes during a multipart upload, you can resume the upload by using the [ListMultipartUploads](#) and [ListParts](#) APIs to retrieve all multipart upload tasks on an object and list the uploaded parts in each task. This allows an upload task to be resumed from the last uploaded part. The same principles apply if you pause and then resume an upload.

Operating method	Description
Java SDK	SDK demos in various languages
Python SDK	
Go SDK	
C SDK	
.NET SDK	
Android SDK	
iOS SDK	

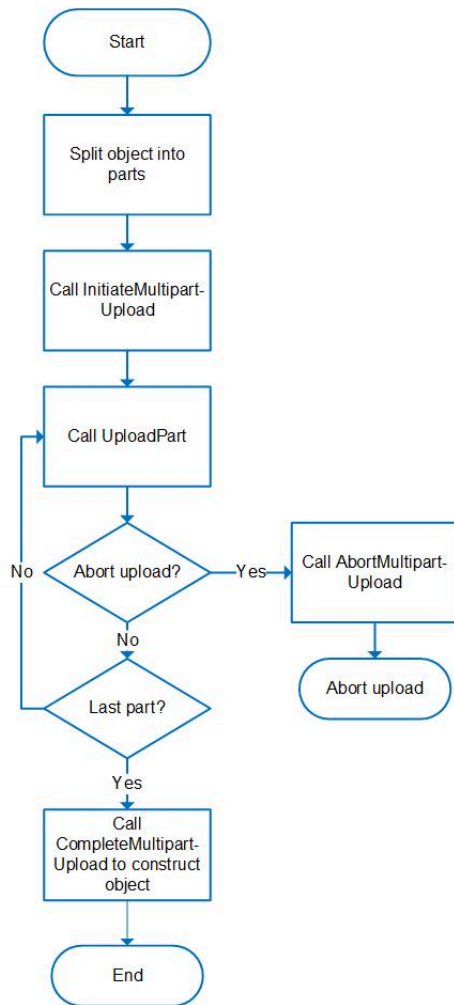
Upload limits

- **Size:** The maximum size of an object is determined by the size of parts. Multipart upload supports a maximum of 10,000 parts. Each part must be at least 100 KB (except for the last part, which may be smaller) and no more than 5 GB. Therefore, the object size cannot exceed 48.8 TB.
- **Naming rules**
 - Object names must be UTF-8 encoded.
 - Object names must be one byte to 1,023 bytes in length.
 - Object names cannot start with a forward slash (/) or a backslash (\).

Multipart upload process

The multipart upload process is as follows:

1. Split the object to be uploaded into multiple parts.
2. Initialize a multipart upload task (through the [InitiateMultipartUpload](#) API).
3. Upload the parts one by one or concurrently (through the [UploadPart](#) API).
4. Complete the upload (through the [CompleteMultipartUpload](#) API).



During the multipart upload process, you need to note the following items:

- Each part except the last one cannot be smaller than 100 KB. Otherwise, you may fail to call the [CompleteMultipartUpload](#) API.
- After the object to be uploaded is split into parts, they are sorted by partNumber specified during the upload. However, because upload in sequence is not required, the parts can be uploaded concurrently.

Due to network conditions and the device load, the upload does not necessarily speed up when more parts are uploaded concurrently. We recommend that you increase the part size in good network conditions, otherwise decrease the part size.

- By default, when all parts are uploaded but you have not called the [CompleteMultipartUpload](#) API, the uploaded parts are not deleted automatically. You can call the [AbortMultipartUpload](#) API to terminate the upload and delete the parts that occupy the storage space. For more information about how to automatically delete the uploaded parts, see [Manage object lifecycle](#).

Upload security and authorization

To prevent unauthorized third-party users from uploading data to your bucket, OSS provides bucket- and object-level access control. For more information, see [Access control](#).

To authorize third-party users to upload objects, OSS also provides account authorization. For more information, see [Authorized third-party upload](#).

Subsequent operations

- After uploading objects to OSS, you can use [upload callback](#) to submit a callback request to the specified application server and perform subsequent operations.
- After uploading images, you can use [Image Processing](#).
- After uploading audio or video objects, you can use [ApsaraVideo for Media Processing](#).

API reference

- Multipart upload APIs:
 - [MultipartUpload](#)
 - [InitiateMultipartUpload](#)
 - [UploadPart](#)
 - [UploadPartCopy](#)
 - [CompleteMultipartUpload](#)
 - [AbortMultipartUpload](#)
 - [ListMultipartUploads](#)
 - [ListParts](#)

9.4 Append upload

By using append upload, you can use the AppendObject API of OSS to directly append content to the appendable objects that are uploaded.



Note:

For more information about the AppendObject API, see [AppendObject](#).

Scenarios

By using [simple upload](#), [form upload](#), and [multipart upload and resumable upload](#), you can only create normal objects that have fixed content after the upload is completed. They can only be read but cannot be modified. To modify the object content, you must upload an object with the same name to overwrite the existing one. This is a major difference between OSS and typical file systems.

Due to this feature, these upload methods are inconvenient in many scenarios, such as video surveillance and video live streaming, because video data is constantly produced in real time. By using these upload methods, you must split a video stream into small parts and then upload them as new objects. In actual use, these methods have obvious defects:

- The software architecture is complex. You must consider intricate issues such as object splitting.
- You must reserve space to store metadata, such as the list of created objects. After receiving each request, OSS must repeatedly read the metadata and determine whether to create an object. This puts high pressure on the server. In addition, the client must send each request twice, causing a certain latency.
- If object parts are small, the latency is low. However, too many objects are hard to manage. If object parts are large, the latency is high.

To simplify development and reduce costs in such scenarios, OSS provides append upload (through the `AppendObject` API), which allows you to directly append content to the end of an object. Objects uploaded by using this method are appendable objects, whereas objects uploaded by using other methods are normal objects. The appended data is instantly readable.

By using append upload, the architecture becomes simple in the preceding scenarios. When video data is produced, it is immediately added to the same object by using append upload. The client only needs to regularly retrieve the object length and compare it with the previous value. If new readable data is found, the client starts a read operation to retrieve the newly uploaded data. This method greatly simplifies the architecture and enhances the scalability.

In addition to video scenarios, append upload can be used to append log data.

Operating methods

Operating method	Description
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Android SDK	
iOS SDK	

Upload limits

- **Size:** The maximum size of an object is 5 GB in this mode.
- **Naming rules:**
 - Object names must be UTF-8 encoded.
 - Object names must be one byte to 1,023 bytes in length.
 - Object names cannot start with a forward slash (/) or a backslash (\).
- **Object type:** You can append data only to objects created by using append upload. Therefore, new data cannot be appended to objects created by using simple upload, form upload, multipart upload, or resumable upload.
- **Subsequent operations:** You cannot copy objects created by using append upload. However, you can modify the Object Meta of the objects.

Upload security and authorization

To prevent unauthorized third-party users from uploading data to your bucket, OSS provides bucket- and object-level access control. For more information, see [Access control](#).

To authorize third-party users to upload objects, OSS also provides account authorization. For more information, see [Authorized third-party upload](#).

Subsequent operations

- After uploading images, you can use [Image Processing](#).

- After uploading audio or video objects, you can use [ApsaraVideo for Media Processing](#).

**Note:**

Append upload does not support upload callback.

9.5 Authorized third-party upload

This topic describes how to authorize a third-party user to upload objects directly to OSS without the need to forward objects through the server.

Scenarios

In a standard client/server system architecture, the server is responsible for receiving and processing requests from the client. If OSS is used as a back-end storage service, the client sends objects to be uploaded to the application server, which then forwards them to OSS. In this process, the data needs to be transmitted twice, once from the client to the server and once from the server to OSS. In the case of bursts of access requests, the server must provide sufficient bandwidth resources for multiple clients to upload objects simultaneously. This presents a challenge to the architecture scalability.

To resolve this issue, OSS provides authorized third-party upload. By using this feature, each client can upload objects directly to OSS without transmitting them to the server. This reduces the cost for application servers and maximizes the OSS capability to process large amounts of data. In this case, you can focus on your business, without worrying about the bandwidth and concurrency limits.

Currently, OSS supports two methods to grant upload permissions: signed URL and temporary access credential.

Signed URL

In this method, you can use a request URL that contains the OSSAccessKeyId and Signature fields to directly upload objects. Each signed URL has expiration time to ensure security.

- Operating methods

Operating method	Description
Java SDK	SDK demos in various languages
Python SDK	

Operating method	Description
PHP SDK	
Go SDK	
C SDK	
.NET SDK	

- For more information, see [Add a signature to a URL](#).

Temporary access credential

Alibaba Cloud uses Security Token Service (STS) to grant temporary access credentials to authorize users. STS is a Web service that provides temporary access tokens for cloud computing users. Through STS, you can grant a third-party application or a RAM user (who you can manage its user identity) an access credential with a custom validity period and permissions.

- Operating methods

Operating method	Description
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Android SDK	
iOS SDK	

- For more information about operation examples, see [Access OSS with a temporary access token provided by STS](#).

9.6 Upload callback

After an object is uploaded, OSS can start a callback process for the application server. To request callback, you only need to send a request that contains relevant callback parameters to OSS.



Note:

- For more information about how to use OSS APIs to implement callbacks, see [Callback](#).
- APIs that support callbacks include [PutObject](#), [PostObject](#), and [CompleteMultipartUpload](#).

Scenarios

Upload callback is typically used together with authorized third-party uploads. When uploading an object to OSS, the client requests a callback to the server. After completing the upload task of the client, OSS automatically sends an HTTP request for the callback to the application server to notify the server that the upload is completed. Then, the server performs operations such as modifying the database and responds to the callback request. After receiving the response, OSS returns the upload status to the client.

When sending a POST callback request to the application server, OSS includes parameters in the POST request body to carry specific information. Such parameters are divided into two types: system-defined parameters such as the bucket name and the object name, and user-defined parameters. You can specify user-defined parameters based on the application logic when sending a request that contains callback parameters to OSS. You can use user-defined parameters to carry information relevant to the application logic, such as the ID of the user who sends the request. For more information about how to use user-defined parameters, see [Callback](#).

You can properly use upload callback to simplify the client logic and save network resources. The following figure shows the process.

**Note:**

- The following regions support upload callback: regions in Mainland China, Hong Kong, Singapore, Australia (Sydney), US (Virginia), US (Silicon Valley), Japan (Tokyo), Germany (Frankfurt), and UAE (Dubai).
- Currently, only simple upload (through the PutObject API), form upload (through the PostObject API), and multipart upload (through the CompleteMultipartUpload API) support upload callback.

Operating methods

Operating method	Description
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
C SDK	
.NET SDK	

Reference

- [Upload callback errors and troubleshooting](#)
- [Direct data transfer on Web clients and upload callback](#)
- [Set up upload callback for mobile applications](#)

9.7 RTMP-based stream ingest

OSS allows you to use Real-Time Messaging Protocol (RTMP) to ingest H.264-encoded video streams and Advanced Audio Coding (AAC)-encoded audio streams to OSS. Audio and video data uploaded to OSS can be played on demand or be used for live streaming in latency-insensitive scenarios.

When uploading audio and video data to OSS in compliance with RTMP, you need to note the following limits:

- By using RTMP, you can only ingest video or audio streams but not pull the streams.
- You must ingest video streams, which are in H.264 format.
- Audio streams are optional, which must be in AAC format. OSS discards audio streams in other formats.
- You can only use HTTP Live Streaming (HLS) to dump audio and video data.
- A LiveChannel can receive streams ingested from only one client at a time.

The following sections describe how to ingest audio and video streams to OSS and how to play the uploaded audio and video data for video-on-demand (VOD) playback and live streaming.

Ingest audio and video streams to OSS

- Obtain an ingest URL

Use the SDK to call the PutLiveChannel API, create a LiveChannel, and obtain the corresponding ingest URL. If the bucket ACL is set to public-read-write, you can directly use the obtained ingest URL. Otherwise, add a signature to the ingest URL.

The following code uses the Python SDK as an example to show how to obtain ingest URLs without and with a signature, respectively.

```
from oss2 import *
from oss2.models import *
host = "oss-cn-hangzhou.aliyuncs.com" # just for example
accessid = "your-access-id"
accesskey = "your-access-key"
bucket_name = "your-bucket"
channel_name = "test-channel"
auth = Auth(accessid, accesskey)
bucket = Bucket(auth, host, bucket_name)
channel_config = LiveChannelInfo(target=LiveChannelInfoTarget())
channel = bucket.create_live_channel(channel_name, channel_config)
publish_url = channel.publish_url
signed_publish_url = bucket.sign_rtmp_url("test-channel", "playlist.m3u8", 3600)
```

The following example shows the obtained ingest URLs:

```
publish_url = rtmp://your-bucket.oss-cn-hangzhou.aliyuncs.com/live/test-channel
signed_publish_url = rtmp://your-bucket.oss-cn-hangzhou.aliyuncs.com/live/your-channel?OSSAccessKeyId=LGarxxxxxx&HjKWg6&playlistName=t.m3u8&Expires =
```



```
1472201595 & Signature = bjKraZTTyz z9 % 2FpYoomDx4 Wgh % 2F1M % 3D "
```

- Use FFmpeg for stream ingest

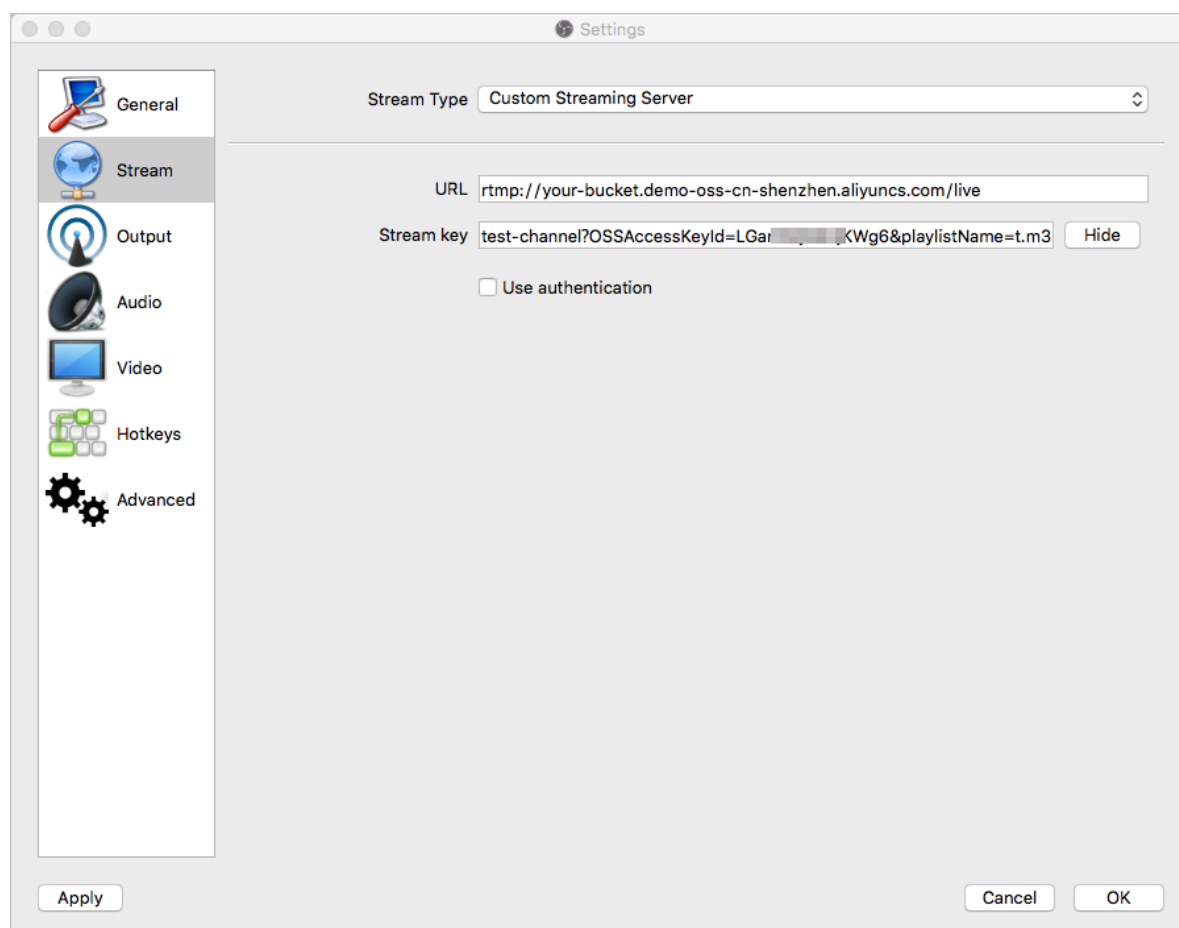
You can use FFmpeg to upload local video files to OSS by running the following command:

```
ffmpeg -i 1.flv -c copy -f flv "rtmp://your-bucket.oss-cn-hangzhou.aliyuncs.com/live/test-channel?OSSAccessKeyId=LGarxxxxxxHjKWg6&Expires=1472199095&Signature=%2FAvRo7FTs%2Fs1InBKgwn7Gz%2FULp9w%3D"
```

- Use OBS for stream ingest

Click Settings. In the URL field, enter the ingest URL that you obtain in the preceding step, and click OK.

As shown in the following figure, you need to note how the ingest URL is split.



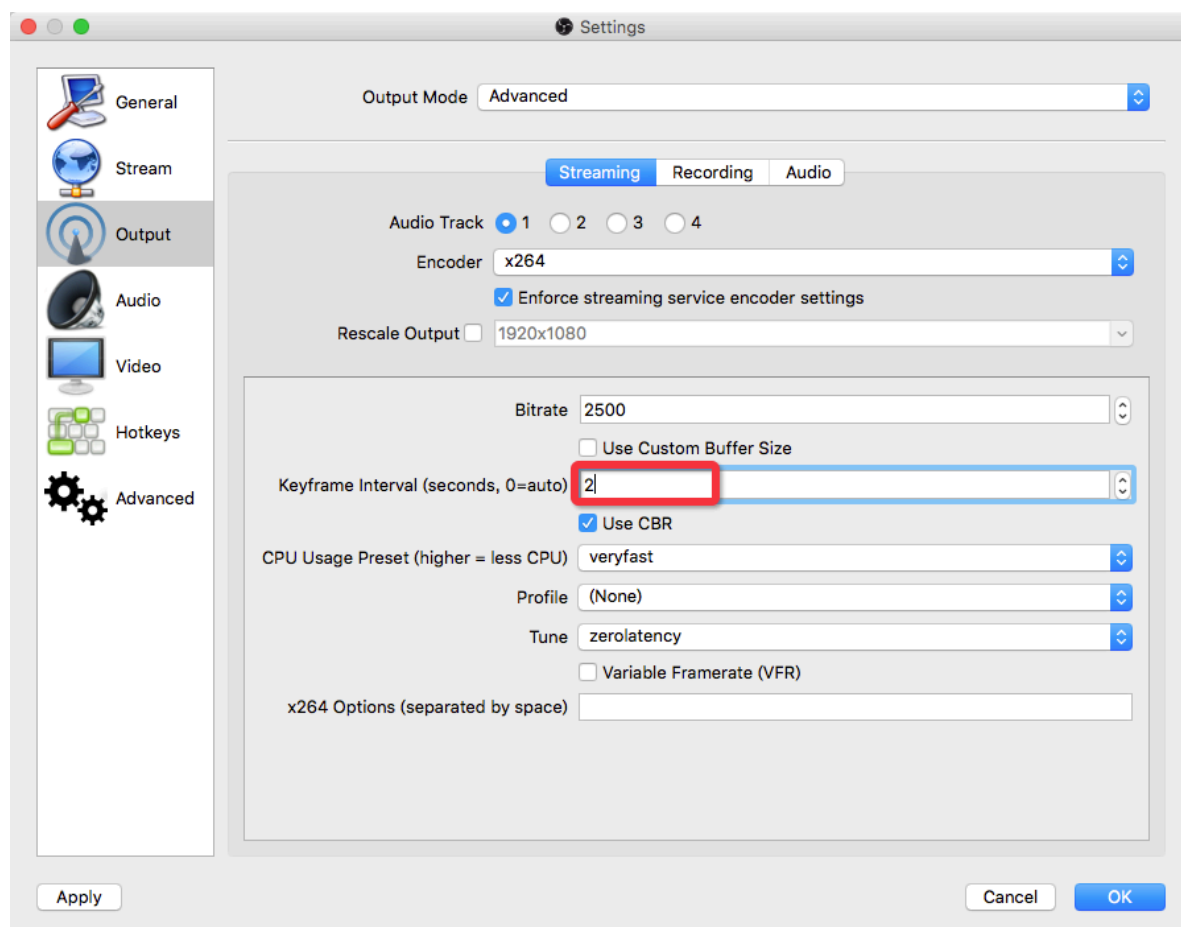
Play the audio and video data uploaded to OSS

- Live streaming

During stream ingest, you can use HLS to play the audio and video data that is being uploaded on the following platforms in different ways:

- On mobile platforms such as Android and iOS, enter the corresponding streaming URL of the LiveChannel in the browser.
- On the macOS platform, use Safari to play the content.
- On a PC, install the VLC media player to play the content.

To smoothly play the uploaded audio and video data for live streaming, you can set `FragDuration` to a small value, for example, 2s. You can also set the group of pictures (GOP) to a fixed value, which is the same as that of `FragDuration` of the LiveChannel. The following figure shows how to set the GOP (that is, `Keyframe Interval`) in OBS.



- **VOD playback**

During stream ingest, OSS always uses live streams to push or update M3U8 objects. In VOD playback scenarios, you must call the PostVodPlaylist API after stream ingest to assemble an M3U8 object for VOD playback and use the object URL to play the uploaded audio and video data.

In VOD playback scenarios, you can set a larger GOP to reduce the number of TS objects and the bit rate.

10 Download files

10.1 Simple download

By using simple download, you can send an HTTP GET request to download an uploaded object through the `GetObject` API of OSS.



Note:

- For more information about the `GetObject` API, see [GetObject](#).
- For more information about the rules for generating object URLs, see [OSS request process](#).
- For more information about how to use a custom domain to access an object, see [Bind a custom domain](#).

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
ossbrowser	Graphical tool, which is easy to operate
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
Android SDK	
iOS SDK	
Node.js SDK	
Browser.js SDK	
Ruby SDK	

Download security and authorization

- To prevent unauthorized third-party users from downloading data from your bucket, OSS provides bucket- and object-level access control. For more information, see [Access control](#).
- For more information about how to authorize a third-party user to download objects from a private bucket, see [Authorized third-party download](#).

Reference

For more information about how to use resumable download, see [Resumable download](#).

10.2 Resumable download

OSS allows you to download data from a specified position of an object. When downloading a large object, you can split it into multiple parts and download them at different points of time. If a download is paused or interrupted, you can also resume it at the paused or interrupted position.

Similar to simple upload, you must have the read permission on the object. You can set the Range parameter to enable resumable download. We recommend that you use this feature to download large objects. For more information about the Range parameter, see the relevant RFC. If the Range parameter is specified in the request header, the response contains the length of the entire object and the range returned in this response. For example, Content-Range: bytes 0–9/44 indicates that the length of the entire object is 44 bytes, and the range returned this time is the first 10 bytes. If the specified Range parameter value is invalid, the entire object is transmitted. The response does not include Content-Range, but returns HTTP status code 206.

Operating methods

Operating method	Description
Java SDK	SDK demos in various languages
Python SDK	
Go SDK	
C SDK	
Android SDK	
iOS SDK	

Download security and authorization

- To prevent unauthorized third-party users from downloading data from your bucket, OSS provides bucket- and object-level access control. For more information, see [Access control](#).
- For more information about how to authorize a third-party user to download objects from a private bucket, see [Authorized third-party download](#).

10.3 Authorized third-party download

To authorize a third-party user to download objects from a private bucket, you must use a signed URL or a temporary access credential. You cannot directly provide the AccessKey.

Signed URL

OSS allows users to use a signed URL to download data. You can add a signed URL and forward the URL to a third-party user to authorize access. The third-party user can then send an HTTP GET request and use the URL to download objects.

- Implementation method

The following example shows how to generate a signed URL.

```
http ://< bucket >.< region >. aliyuncs . com /< object >?  
OSSAccessK eyId =< user   access_key _id >& Expires =< unix  
time >& Signature =< signature_ string >
```

A signed URL must include at least the following three parameters: Signature, Expires, and OSSAccessKeyId.

- OSSAccessKeyId: The AccessKey ID of your Alibaba Cloud account.
- Expires: The expected expiration time of the URL.
- Signature: The signature string. For more information, see [Add a signature to a URL](#).



Note:

This link must be URL-encoded.

- Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	

Temporary access credential

OSS uses Security Token Service (STS) to provide temporary credentials for third-party users. By adding a signature to the request header, a third-party user can access objects. This authorization method is applicable to object download in mobile scenarios. For more information about how to implement temporary access credentials, see [STS Java SDK](#).

· Implementation method

A third-party user sends a request to the application server to obtain the AccessKey ID, AccessKey Secret, and STS Token issued by STS. The user then uses the obtained AccessKey ID, AccessKey Secret, and STS Token to request the developer's object resources.

· Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	

Operating method	Description
C SDK	
.NET SDK	
Android SDK	
iOS SDK	

Best practices

- [RAM and STS User Guide](#)

10.4 OSS Select

By using OSS Select, you can use simple SQL statements to select content from objects in OSS to obtain only required data. OSS Select helps you reduce the amount of data transmitted from OSS to improve the data retrieval efficiency and save time.



Note:

You can send requests to transmit SQL expressions to OSS. For more information about SQL statements supported by OSS Select and limits, see [SelectObject](#).

Operating methods

OSS SDK (Currently, Java and Python SDKs support OSS Select.)

Limits

- **Object format:** OSS Select supports CSV objects that are UTF-8 encoded and conform to RFC 4180, including CSV-like objects such as TSV objects. In the objects, row and column delimiters and quote characters can be customized.
- **Encryption method:** OSS Select supports objects encrypted by using server-side encryption fully managed by OSS (SSE-OSS) or server-side encryption that uses customer master keys (CMKs) managed by Key Management Service (KMS) for encryption (SSE-KMS).
- **Region:** OSS Select is currently available only in China (Shenzhen).

11 Manage files

11.1 Manage Object Meta

Object Meta describes the attributes of objects uploaded to OSS. These attributes are classified into two types: HTTP standard attributes (HTTP header fields) and User Meta (user-defined Object Meta). You can set Object Meta when uploading or copying objects in various ways.

- HTTP standard attributes

Name	Description
Cache-Control	The Webpage caching behavior when the object is downloaded.
Content-Disposition	The name of the object during the download.
Content-Encoding	The content encoding format of the object during the download.
Content-Language	The content language encoding when the object is downloaded.
Expires	The time the object expires.
Content-Length	The size of the object.
Content-Type	The type of the object.
Last-Modified	The time the object is last modified.

- User Meta

User Meta allows you to better describe objects. In OSS, all parameters prefixed with x-oss-meta- are considered as User Meta, such as x-oss-meta-location. An object may have multiple similar parameters, but the total size of all User Meta cannot exceed 8 KB. User Meta is returned in the HTTP header of responses to GetObject or HeadObject requests.

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use

Operating method	Description
ossbrowser	Graphical tool, which is easy to operate
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Android SDK	

Reference

You can also add Object Meta in the following operations:

- [Simple upload](#)
- [Multipart upload and resumable upload](#)
- [Append upload](#)
- [Copy objects](#)

11.2 View the object list

You can use the GetBucket API of OSS to list the objects that you upload in a bucket.



Note:

For more information about the GetBucket API, see [GetBucket](#).

You can call the GetBucket API to view the list of up to 1,000 objects in a bucket at a time. The following table describes the parameters that you can use to list objects in various ways.

Name	Description
Delimiter	The character used to group objects by name. If you specify the Delimiter parameter in the request, the response contains the CommonPrefixes element. This element indicates a set of object names that share a specified prefix and end with the first specified delimiter.
Marker	The starting position of the object list. Objects are sorted in alphabetical order and those objects following this marker are listed.
MaxKeys	The maximum number of objects that are returned. The default value is 100, and the maximum value is 1000.
Prefix	The prefix that must be contained in the names (key) of the returned objects. Note that if you use a prefix to query objects, the returned key values still contain the prefix.

Operating methods

Operating method	Description
Console	Web application that lists the objects in a bucket on the Files tab of the bucket overview page
ossbrowser	Graphical tool, which is easy to operate
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Node.js SDK	
Browser.js SDK	
Ruby SDK	

Folder simulation

OSS does not support folders. All elements are stored as objects. To simulate a folder, you can create an empty object whose name ends with a forward slash (/). This object can be uploaded and downloaded. The console displays any object whose name ends with a forward slash (/) as a folder. Therefore, you can use the preceding method to create a simulated folder.

You can use the `Delimiter` and `Prefix` parameters together to simulate folders as follows:

- If you set the `Prefix` parameter to a folder name in the request, the objects whose names start with this prefix are listed, including all recursive objects and subfolders (directories) in this folder. The object names are listed in the `Contents` element.
- If you also set the `Delimiter` parameter to a forward slash (/) in the request, the objects whose names start with the specified prefix and subfolders (directories) in the folder are listed. The subfolders (directories) are listed in the `CommonPrefixes` element, excluding recursive objects and folders in these subfolders.

Example:

The OSS bucket `oss - sample` contains the following objects:

```
Object    D
Directory A / Object    C
Directory A / Object    D
Directory A / Directory B / Object    B
Directory A / Directory B / Directory C / Object    A
Directory A / Directory C / Object    A
Directory A / Directory D / Object    B
Directory B / Object    A
```

- List level-1 directories and objects

Based on the API request conventions, leave the `Prefix` parameter empty and set the `Delimiter` parameter to a forward slash (/). The response is as follows:

```
<? xml version=" 1 . 0 " encoding=" UTF - 8 "? >
< ListBucket Result >
  < Name > oss - sample </ Name >
  < Prefix ></ Prefix >
  < Marker ></ Marker >
  < MaxKeys > 1000 </ MaxKeys >
  < Delimiter >/</ Delimiter >
  < IsTruncate d > false </ IsTruncate d >
  < Contents >
    < Key > Object    D </ Key >
```

```

    < LastModifi ed > 2015 - 11 - 06T10 : 07 : 11 . 000Z </
LastModifi ed >
    < ETag >" 8110930DA5 E04B1ED5D8 4D6CC4DC90 80 "</ ETag >
    < Type > Normal </ Type >
    < Size > 3340 </ Size >
    < StorageCla ss > Standard </ StorageCla ss >
    < Owner >
      < ID > oss </ ID >
      < DisplayNam e > oss </ DisplayNam e >
    </ Owner >
  </ Contents >
  < CommonPref ixes >
    < Prefix > Directory A </ Prefix >
  </ CommonPref ixes >
  < CommonPref ixes >
    < Prefix > Directory B </ Prefix >
  </ CommonPref ixes >
</ ListBucket Result >

```

In the preceding response:

The Contents element contains the level-1 object: Object D. The CommonPrefixes element contains the level-1 directories: Directory A/ and Directory B/, but does not list the objects in these directories.

- List level-2 directories and objects in Directory A

Based on the API request conventions, set the Prefix parameter to Directory A and the Delimiter parameter to a forward slash (/). The response is as follows:

```

<? xml version =" 1 . 0 " encoding =" UTF - 8 "? >
< ListBucket Result >
  < Name > oss - sample </ Name >
  < Prefix > Directory A </ Prefix >
  < Marker ></ Marker >
  < MaxKeys > 1000 </ MaxKeys >
  < Delimiter >/</ Delimiter >
  < IsTruncate d > false </ IsTruncate d >
  < Contents >
    < Key > Directory A / Object C </ Key >
    < LastModifi ed > 2015 - 11 - 06T09 : 36 : 00 . 000Z </
LastModifi ed >
    < ETag >" B026324C69 04B2A9CB4B 88D6D61C81 D1 "</ ETag >
    < Type > Normal </ Type >
    < Size > 2 </ Size >
    < StorageCla ss > Standard </ StorageCla ss >
    < Owner >
      < ID > oss </ ID >
      < DisplayNam e > oss </ DisplayNam e >
    </ Owner >
  </ Contents >
  < Contents >
    < Key > Directory A / Object D </ Key >
    < LastModifi ed > 2015 - 11 - 06T09 : 36 : 00 . 000Z </
LastModifi ed >
    < ETag >" B026324C69 04B2A9CB4B 88D6D61C81 D1 "</ ETag >
    < Type > Normal </ Type >
    < Size > 2 </ Size >
    < StorageCla ss > Standard </ StorageCla ss >
    < Owner >

```

```

    < ID > oss </ ID >
    < DisplayNam e > oss </ DisplayNam e >
  </ Owner >
</ Contents >
< CommonPref ixes >
  < Prefix > Directory    A / Directory    B </ Prefix >
</ CommonPref ixes >
< CommonPref ixes >
  < Prefix > Directory    A / Directory    C </ Prefix >
</ CommonPref ixes >
< CommonPref ixes >
  < Prefix > Directory    A / Directory    D </ Prefix >
</ CommonPref ixes >
</ ListBucket Result >

```

In the preceding response:

The Contents element contains the level-2 objects: Directory A/Object C and Directory A/Object D. The CommonPrefixes element contains the level-2 directories: Directory A/Directory B/, Directory A/Directory C/, and Directory A/Directory D/, but does not list the objects in these directories.


11.3 Copy objects

You can copy objects from a bucket to another bucket without modifying the object content.

Previously, to copy an object, you need to download it and then upload it to the destination bucket. This method wastes network bandwidth resources. Therefore, OSS provides the CopyObject API to copy objects within OSS. You do not need to transmit large amounts of data from and to OSS.

In addition, OSS does not allow you to rename objects. We recommend that you call the CopyObject API to first copy the data of an object to another object with a new name, and then delete the original object. To modify only the Object Meta of an object, you can also call the CopyObject API and set the source address and the destination address to the same value. In this way, OSS only updates Object Meta. For more information about Object Meta, see [Manage Object Meta](#).

Operating methods

Operating method	Description
ossbrowser	Graphical tool, which is easy to operate <div>  Notice: You can only use ossbrowser to copy objects smaller than 5 GB. </div>
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Node.js SDK	
Browser.js SDK	
Ruby SDK	

Notes

When copying objects, you need to note the following items:

- You must have operation permissions on the source object. Otherwise, the operation may fail.
- You are not allowed to copy data across regions. For example, you are not allowed to copy an object in a bucket in China (Hangzhou) to a bucket in China (Qingdao).
- You are not allowed to copy objects created by using append upload.
- You can use the [CopyObject](#) API to copy objects smaller than 1 GB in simple copy mode.
- You can use the [UploadPartCopy](#) API to copy objects larger than 1 GB in multipart copy mode.

11.4 Delete objects

You can delete objects that are uploaded to OSS buckets.

OSS allows you to perform the following deletion operations:

- **Single deletion:** You can delete a specific object.
- **Batch deletion:** You can delete up to 1,000 objects at a time.
- **Automatic deletion:** We recommend that you [manage object lifecycle](#) to enable automatic object deletion if you need to delete a large number of objects based on certain rules, for example, you need to regularly delete objects that were created certain days ago or empty a bucket. After you set lifecycle management rules, OSS automatically deletes expired objects based on the rules. This greatly reduces the number of deletion requests that you send and improves the deletion efficiency.

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
ossbrowser	Graphical tool, which is easy to operate
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Node.js SDK	
Browser.js SDK	
Ruby SDK	

11.5 Manage back-to-origin configurations

After you set back-to-origin rules, OSS retrieves requested data from the origin in multiple ways to meet your requirements such as hot data migration and specific request redirection.



Note:

For more information about how to set back-to-origin rules, see [Set back-to-origin rules](#).

OSS matches the URL of each GET request based on the preset rules, and retrieves the requested data from the origin accordingly. You can set a maximum of five rules. OSS matches requests based on the rules in sequence until a valid rule is matched. Mirroring back-to-origin and redirection back-to-origin are available.

Mirroring



The process is as follows: If a client sends a GET request to request an object that does not exist in OSS, OSS requests the object from the origin. After obtaining the object, OSS returns it to the client and stores it to process subsequent requests.

Scenarios

Mirroring back-to-origin is used to seamlessly migrate data to OSS. You can migrate any service that is already running in a self-built origin or in another cloud product to OSS without service interruption. Specific scenarios are as follows:

- The origin has an amount of cold data and is constantly generating hot data.

You can use the migration tool [OssImport](#) to migrate cold data to OSS and configure mirroring back-to-origin to set the origin URL in OSS. Even if some newly generated data is not migrated when you switch the domain of your application to OSS (or Alibaba Cloud CDN, which retrieves data from OSS), you can still access the data through OSS. The data is stored in OSS after the first access. After the domain is switched, the origin no longer generates data. You can scan the origin again and

import data that is still not migrated to OSS at a time. Then, you can delete the mirroring back-to-origin configuration to complete data migration.

If the configured origin URL is an IP address, OSS can still retrieve requested data from the origin after you switch the domain of your application to OSS. However, if the configured origin URL is a domain, mirroring back-to-origin does not work because the domain is resolved to OSS or CDN. In this case, you can apply for another domain as the origin and configure this domain to be resolved to the same IP address as the serving domain. In this way, after you switch the serving domain to OSS, OSS can still retrieve requested data from the origin.

- Only some traffic of the origin is switched to OSS or CDN, and the origin continues to generate data.

The migration method is similar to that described in the preceding scenario. After switching a portion of traffic to OSS, you do not need to delete the mirroring back-to-origin configuration. This ensures that OSS can still retrieve requested data from the origin for the traffic switched to OSS or CDN.

Detail analysis

- OSS uses mirroring back-to-origin to request an object from the origin only when `GetObject()` returns HTTP status code 404.
- OSS uses `MirrorURL + object` as the back-to-origin URL to request data from the origin, where `MirrorURL` indicates the origin URL and `object` indicates the name of the requested object. For example, you configure mirroring back-to-origin for the bucket `example - bucket`, with `MirrorURL` set to `http://www.example-domain.com/`. The bucket does not include the object `image/example_object.jpg`. If a user needs to download this object, OSS sends a GET request to obtain the object from `http://www.example-domain.com/image/example_object.jpg`, stores the obtained object, and returns the object to the user. After the object is downloaded, it is available in OSS as `image/example_object.jpg`. This process functions the same as that of migrating an object with the same name from the origin to OSS. If `MirrorURL` carries path information, such as `http://www.example-domain.com/dir1/`, the process is the same as that in the preceding example. The back-to-origin URL is `http://www.example-domain.com/dir1/image/example_object.jpg`. OSS also obtains the object `image/example_ob`

`ject . jpg` . This process functions the same as that of migrating an object from a directory of the origin to OSS.

- OSS does not send the header information transmitted by a user to the origin. Whether OSS sends the QueryString information to the origin depends on the back-to-origin configuration in the console.
- If the origin returns chunked-encoded data, OSS also returns chunked-encoded data to the user.
- OSS returns to the user and stores the following header information received from the origin:

```
Content - Type
Content - Encoding
Content - Dispositio n
Cache - Control
Expires
Content - Language
Access - Control - Allow - Origin
```

- When returning objects obtained through mirroring back-to-origin, OSS adds the `x-oss-tag` field to the response header and sets its value to `MIRROR + Space + url_decode (back-to-origin URL)`. The following code provides an example.

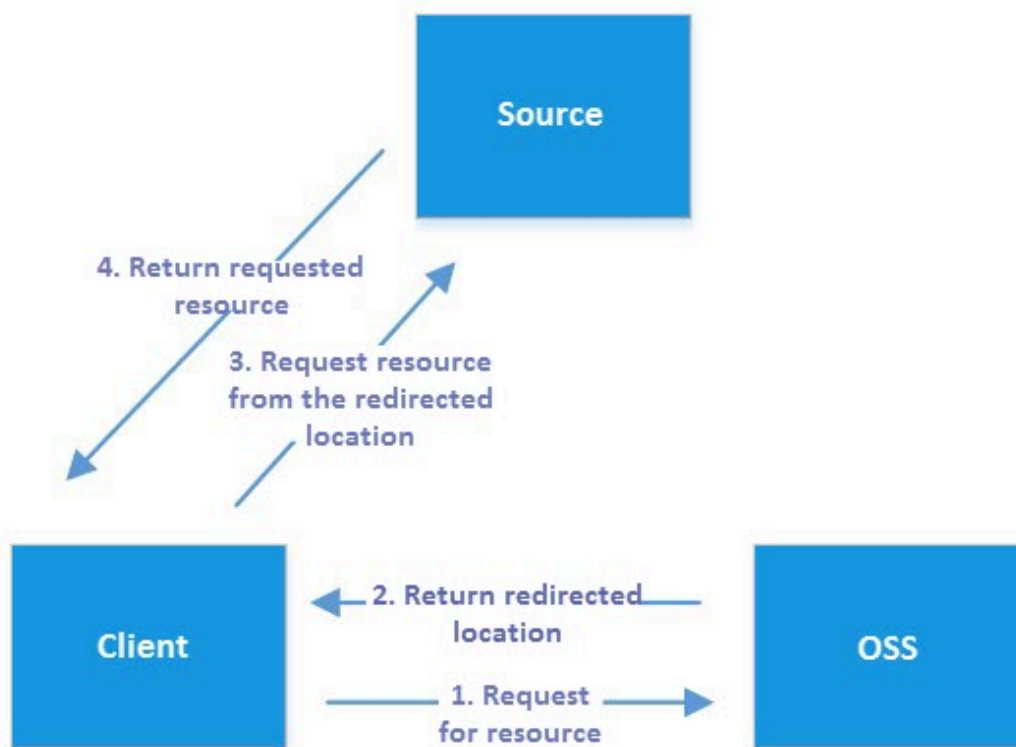
```
x - oss - tag : MIRROR http % 3a % 2f % 2fwww . example - domain
. com % 2fdir1 % 2fimage % 2fexample_ object . jpg
```

After OSS obtains an object from the origin, so long as it is not overwritten, OSS adds the `x-oss-tag` field to the response header each time a user downloads the object. This header field indicates that the object is sourced from mirroring back-to-origin.

- After OSS obtains an object through mirroring back-to-origin, if the corresponding object is modified in the origin, OSS does not update the obtained object. In this case, this object is already in OSS and does not meet the mirroring back-to-origin conditions.
- If a requested object does not exist in the origin, the origin returns HTTP status code 404 to OSS, and OSS returns the same code to the user. If the origin returns another non-200 HTTP status code to indicate an error, such as object retrieval failure due to network-related causes, OSS returns HTTP status code 424 to the user, with the error code `MirrorFailed`.

Redirection

The URL redirection feature enables OSS to return a 3XX redirect based on user-defined conditions and corresponding redirection configuration. You can use this redirection feature to redirect objects and use various services accordingly. The following figure shows the process.



Scenarios

- Seamless data migration from other data sources to OSS

A user asynchronously migrates data from the user's own data source to OSS. In this process, if the user's client requests data that is not migrated to OSS, OSS returns a 302 redirect to the client through URL rewriting. Then, the client reads data from the user's data source based on the Location field in the 302 redirect.

- Page redirection

A user wants to hide objects with a certain prefix and return a specific page to visitors.

- Redirect page for a 404 or 500 error

If a 404 or 500 error occurs, users can be redirected to a preset page. This prevents OSS from exposing system errors to users.

11.6 Use the SelectObject API

The SelectObject API is commonly used to analyze logs. It can also be used together with big data products. This topic describes how to use the Python SDK and Java SDK to call the SelectObject API in the preceding scenarios.

Introduction

Object Storage Service (OSS) is a secure and highly reliable cloud storage service based on the Apsara system. It allows the general-purpose storage of large amounts of data on the Internet at a low cost. It also supports RESTful APIs and the auto scaling of capacity and processing capability. OSS can not only store a large number of media objects, but also serve as a data warehouse to store a large number of data objects. Currently, Hadoop 3.0 has supported OSS. You can directly read and process data in OSS when you run services such as Spark, Hive, and Presto in Amazon Elastic MapReduce (EMR) or use Alibaba Cloud services such as MaxCompute, HybridDB, and newly published Data Lake Analytics.

However, the current GetObject API provided by OSS only allows the big data platform to download all OSS data locally for analysis and filtering. As a result, a large number of bandwidth resources and client resources are wasted in many query scenarios.

To solve this problem, OSS provides the SelectObject API. It allows OSS, but not the big data platform, to preliminarily filter data by using conditions and projections and only return useful data to the big data platform. In this way, the client can consume fewer bandwidth resources and process less data to maximize the use of CPU and memory resources. This makes OSS-based data warehousing and data analysis a highly attractive option.

The SelectObject API is currently supported by Java and Python SDKs, and will soon be supported by SDKs in other languages. The SelectObject API supports CSV objects and JSON objects that are UTF-8 encoded. The CSV objects conform to RFC 4180, including CSV-like objects such as TSV objects. In the CSV objects, row and column delimiters and quote characters can be customized. The SelectObject API

supports objects of the Standard and infrequent access (IA) storage classes. (Objects of the Archive storage class must be restored before use.) The SelectObject API also supports objects encrypted by using server-side encryption fully managed by OSS (SSE-OSS) or server-side encryption that uses customer master keys (CMKs) managed by Key Management Service (KMS) for encryption (SSE-KMS).

The SelectObject API supports JSON objects in DOCUMENT and LINES formats. A JSON DOCUMENT object contains a single object. A JSON LINES object is composed of lines of objects separated by delimiters. However, the JSON object itself may not be a valid object. The SelectObject API supports typical delimiters such as `\n` and `\r\n`. You do not need to specify the delimiters.

- Supported SQL syntax
 - SQL statements: SELECT, FROM, and WHERE
 - Data types: string, int64, double64, decimal128, timestamp, and Boolean
 - Operations: logical conditions (AND, OR, and NOT), arithmetic expressions (+, -, ×, /, and %), comparison operations (>, =, <, >=, <=, and !=), and string operations (LIKE and ||)

- Multipart query

The SelectObject API supports multipart query that is similar to byte-based multipart download supported by the GetObject API. Data is divided into parts by row or split. Dividing data by row is commonly used but may result in unbalanced load when sparse data is divided. Dividing data by split is more efficient because the size of each split, which includes multiple rows, is roughly the same.

- Data type

In OSS, data in CSV objects is of the string type by default. You can use the CAST function to convert the data type. For example, the following SQL statement

converts the data in the first and second columns into the integer type and compares them.

```
Select * from OSSObject where cast ( _1 as int ) > cast
( _2 as int )
```

In addition, the SelectObject API allows you to implicitly convert the data type in a WHERE clause. For example, the following SQL statement converts the data in the first and second columns into the integer type.

```
Select _1 from ossobject where _1 + _2 > 100
```

If you do not use the CAST function in an SQL statement, the data type of a JSON object is determined by the data type in the object. A standard JSON object can support data types such as null, Boolean, int64, double, and string.

Python SDK

```
import os
import oss2

def select_callback ( consumed_bytes , total_bytes = None
):
    print ( ' Consumed Bytes : ' + str ( consumed_bytes ) + '\n
' )

# Initialize OSS information such as the AccessKey ID
, AccessKey Secret , and endpoint .
# Obtain the information through environment variables
or replace variables such as < yourAccessKeyId > with
actual values .
#
# Use China ( Hangzhou ) as an example to set the
endpoint to one of the following :
# http://oss-cn-hangzhou.aliyuncs.com
# https://oss-cn-hangzhou.aliyuncs.com

access_key_id = os . getenv ( ' OSS_TEST_ACCESS_KEY_ID ' , '<
yourAccessKeyId >' )
access_key_secret = os . getenv ( ' OSS_TEST_ACCESS_KEY_SECRET
' , '< yourAccessKeySecret >' )
bucket_name = os . getenv ( ' OSS_TEST_BUCKET ' , '< yourBucket
>' )
endpoint = os . getenv ( ' OSS_TEST_ENDPOINT ' , '< yourEndpoint
>' )

# Create an OSS bucket . All object - related methods
must be called through the bucket .
bucket = oss2 . Bucket ( oss2 . Auth ( access_key_id ,
access_key_secret ) , endpoint , bucket_name )
key = ' python_select . csv '
content = ' Tom Hanks , USA , 45 \r \n ' * 1024
filename = ' python_select . csv '
# Upload a CSV object .
bucket . put_object ( key , content )
# Set the parameters of the SelectObject API .
```

```

csv_meta_params = {'CsvHeaderInfo': 'None',
'RecordDelimiter': '\r\n'}
select_csv_params = {'CsvHeaderInfo': 'None',
'RecordDelimiter': '\r\n',
'LineRange': (500, 1000)}

csv_header = bucket.create_select_object_meta(key,
csv_meta_params)
print(csv_header.rows)
print(csv_header.splits)
result = bucket.select_object(key, "select * from
ossobject where _3 > 44", select_cal_l_back, select_csv
_params)
select_content = result.read()
print(select_content)

result = bucket.select_object_to_file(key, filename,
"select * from ossobject where _3 > 44", select_cal
l_back, select_csv_params)
bucket.delete_object(key)

### JSON DOCUMENT
key = 'python_select.json'
content = "{\"contacts\": [{\"key1\": 1, \"key2\": \"hello
world1\"}, {\"key1\": 2, \"key2\": \"hello world2\"}]}"
filename = 'python_select.json'
# Upload a JSON DOCUMENT object.
bucket.put_object(key, content)
select_json_params = {'JsonType': 'DOCUMENT'}
result = bucket.select_object(key, "select s.key2
from ossobject.contacts [*] s where s.key1 = 1", None
, select_json_params)
select_content = result.read()
print(select_content)

result = bucket.select_object_to_file(key, filename,
"select s.key2 from ossobject.contacts [*] s where s.
key1 = 1", None, select_json_params)

bucket.delete_object(key)
### JSON LINES
key = 'python_select_lines.json'
content = "{\"key1\": 1, \"key2\": \"hello world1\"}\n {\"
key1\": 2, \"key2\": \"hello world2\"}"
filename = 'python_select.json'
# Upload a JSON LINES object.
bucket.put_object(key, content)
select_json_params = {'JsonType': 'LINES'}
json_header = bucket.create_select_object_meta(key,
select_json_params)
print(json_header.rows)
print(json_header.splits)

result = bucket.select_object(key, "select s.key2
from ossobject s where s.key1 = 1", None, select_jso
n_params)
select_content = result.read()
print(select_content)
result = bucket.select_object_to_file(key, filename,
"select s.key2 from ossobject s where s.key1 = 1
", None, select_json_params)

```



```
bucket . delete_obj  ect ( key )
```

SelectObject API in Python

· select_object

- The following example shows the sample code of the select_object operation.

```
def select_obj  ect ( self , key , sql ,
                    progress_c  allback = None ,
                    select_par  ams = None
                    ):
```

The preceding sample code is used to run an SQL statement on the object with the specified key and return query results.

- The SQL statement can be directly used as the value of the sql parameter and does not need to be Base64-encoded.
- The progress_callback parameter is optional. It specifies a callback function used to report the query progress.
- The select_params parameter is important for the API. It specifies the parameters and actions of the select_object operation.
- You can use the headers parameter to specify the header information included in the request. The header information functions the same as that for the GetObject API. For example, you can set the bytes field in the request header to specify the range that an SQL statement can query in a CSV object.
- The following table describes the parameters supported by the select_params parameter.

Name	Description
Json_Type	<ul style="list-style-type: none"> ■ If this parameter is left empty, the object is a CSV object by default. ■ If this parameter is set to DOCUMENT, the object is a JSON object. ■ If this parameter is set to LINES, the object is a JSON LINES object.

Name	Description
CsvHeaderInfo	<p>The header information of the CSV object. Valid values: None , Ignore , and Use .</p> <ul style="list-style-type: none"> ■ None: indicates that this object does not include header information. ■ Ignore: indicates that this object includes header information, which is not used in the SQL statement. ■ Use: indicates that this object includes header information, and the column names in the header information are used in the SQL statement.
CommentCharacter	The comment character in the CSV object. The parameter value can be only one character. Default value: None, indicating no comment character.
RecordDelimiter	The row delimiter in the CSV object. The parameter value can be only one or two characters. Default value: \ n .
OutputRecordDelimiter	The row delimiter in the output result of the select_object operation. Default value: \ n .
FieldDelimiter	The column delimiter in the CSV object. The parameter value can be only one character. Default value: comma (,).
OutputFieldDelimiter	The column delimiter in the output result of the select_object operation. Default value: comma (,).
QuoteCharacter	The quote character for the columns in the CSV object. The parameter value can be only one character. Default value: double quotation marks ("). Row and column delimiters enclosed in quotation marks are processed as normal characters.
SplitRange	The split range in multipart queries. The parameter value is a closed interval in (start, end) format, indicating that splits from start# to end# are queried .
LineRange	The row range in multipart queries. The parameter value is a closed interval in (start, end) format, indicating that rows from start# to end# are queried .

Name	Description
CompressionType	The compression type. Default value: None. Valid value: GZIP.
KeepAllColumns	<p>If this parameter is set to true, columns that are excluded by the SELECT statement in the CSV object are left empty in the output result. However, the column positions are kept. Default value: false.</p> <p>Example:</p> <p>A CSV object includes the columns firstname, lastname, and age. The SQL statement is <code>select firstname , age from ossobject</code> . If the KeepAllColumns parameter is set to true, the output result is <code>firstname,,age</code>, in which a comma (,) is added to indicate the position of the excluded lastname column. If the KeepAllColumns parameter is set to false, the output result is <code>firstname,age</code>. By using this parameter, you do not need to modify the code, but can directly use the code that is used to process the GetObject API to process the SelectObject API.</p>
OutputRawData	<ul style="list-style-type: none"> ■ If this parameter is set to true, the select_object operation directly returns the original data. If data is not returned for a long time, a timeout error may occur. ■ If this parameter is set to false, the output data is encapsulated in frames. Default value: false.
EnablePayloadCrc	Indicates whether a cyclic redundancy check (CRC) value is calculated for each frame. Default value: false.
OutputHeader	The header information in the first line of the output result. This parameter applies only to CSV objects.

Name	Description
SkipPartialDataRecord	<p>■ If this parameter is set to true, the current record is skipped when a column in a CSV object has no data or a key in a JSON object does not exist.</p> <p>■ If this parameter is set to false, a column without data is left empty in the output result.</p> <p>Example:</p> <p>A row includes the columns firstname, lastname, and age. The SQL statement is <code>select _1 , _4 from ossobject</code> . If the SkipPartialDataRecord parameter is set to true, this row is skipped. If the SkipPartialDataRecord parameter is set to false, the following result is returned: <code>firstname,\n</code>.</p>
MaxSkippedRecordsAllowed	The maximum number of skipped rows. Default value: 0, indicating that an error is returned if a row is skipped.
ParseJsonNumberAsString	<p>If this parameter is set to true, numbers in the JSON object are resolved as strings. If this parameter is set to false, numbers in the JSON object are resolved as integers or floating-point numbers. Default value: false.</p> <p>High-accuracy floating-point numbers in a JSON object suffer from loss of accuracy when they are resolved as floating-point numbers. To keep the accuracy, you can set this parameter to true and use the CAST function to convert the resolved data into the decimal type.</p>

- Returned result of the `select_object` operation: A `SelectObjectResult` object is returned. You can use the `read()` function or the `__iter__` method to obtain all results. If the output result contains large amounts of data, the `read()` function is not the optimal method to obtain all results. This function may block the system until all results are returned and use excessive memory resources.

We recommend that you use the `__iter__` method (foreach chunk in result) to obtain all results and process each chunk in the results. This method uses fewer

memory resources and allows the client to process each chunk immediately after it is processed by the OSS server. The client does not need to wait for all results to be returned.

- `select_object_to_file`

```
def select_object_to_file ( self , key , filename , sql ,
                           progress_callback = None ,
                           select_params = None
                           ):
```

The preceding sample code is used to run an SQL statement on the object with the specified key and write the query results to another specified object.

Other parameters are the same as those of the [select_object](#) operation.

- `create_select_object_meta`

- The following sample code shows the syntax of the `select_meta_params` parameter.

```
def create_select_object_meta ( self , key , select_meta_params = None ):
```

The preceding sample code is used to create Select Meta for the object with the specified key or obtain Select Meta from such an object. Select Meta includes the total number of rows, total number of columns (for CSV objects), and total number of splits in an object.

If Select Meta has been created for an object, this function does not re-create Select Meta unless the value of the `OverwriteIfExists` parameter is set to true.

To create Select Meta for an object, the object must be completely scanned.

- The following table describes the parameters supported by the `select_meta_params` parameter.

Name	Description
Json_Type	If this parameter is left empty, the object is a CSV object by default. If this parameter is specified, the parameter value must be <code>LINES</code> , indicating that the object is a JSON LINES object. This operation does not apply to JSON DOCUMENT objects.
RecordDelimiter	The row delimiter in the CSV object.
FieldDelimiter	The column delimiter in the CSV object.

Name	Description
QuoteCharacter	The quote character in the CSV object. Row and column delimiters enclosed in quotation marks are processed as normal characters.
CompressionType	The compression type. If this parameter is specified, the parameter value must be None.
OverwriteIfExists	Indicates whether the created Select Meta overwrites the original Select Meta. You do not need to set this parameter in common scenarios.

- Returned result of the create_select_object_meta operation: A GetSelectObjectMetaResult object is returned, which includes the rows and splits attributes. For a CSV object, the select_resp object in the result includes the columns attribute, indicating the number of columns in the CSV object.

Java SDK

```

package    samples ;

import    com . aliyun . oss . ClientBuilderConfiguration ;
import    com . aliyun . oss . model . * ;
import    com . aliyun . oss . OSS ;
import    com . aliyun . oss . OSSClientBuilder ;

import    java . io . IOException ;
import    java . util . ArrayList ;
import    java . util . List ;
import    java . util . concurrent . Callable ;
import    java . util . concurrent . ExecutorService ;
import    java . util . concurrent . Executors ;
import    java . util . concurrent . Future ;

import    com . aliyun . oss . common . auth . * ;
import    com . aliyuncs . DefaultAcsClient ;
import    com . aliyuncs . exceptions . ClientException ;
import    com . aliyuncs . http . MethodType ;
import    com . aliyuncs . http . ProtocolType ;
import    com . aliyuncs . profile . DefaultProfile ;
import    com . aliyuncs . profile . IClientProfile ;
import    com . aliyuncs . sts . model . v20150401 . AssumeRoleRequest ;
import    com . aliyuncs . sts . model . v20150401 . AssumeRoleResponse ;

import    java . text . SimpleDateFormat ;
import    java . util . Calendar ;

/**
 * Examples of the create_select_object_meta and
 * select_object operations .
 */
class    MultipartSelector implements Callable < Integer > {

```

```

        private OSS client ;
        private String bucket ;
        private String key ;
        private int start ;
        private int end ;
        private String sql ;

        public MultipartSelector ( OSS client , String bucket ,
String key , int start , int end , String sql ){
            this . client = client ;
            this . bucket = bucket ;
            this . key = key ;
            this . start = start ;
            this . end = end ;
            this . sql = sql ;
        }

        @Override
        public Integer call () throws Exception {
            SelectObjectRequest selectObjectRequest =
                new SelectObjectRequest ( bucket , key )
                    . withInputStream ( new InputSerialization ()
                        . withCsvInputFormat (
                            new CSVFormat ()
                                . withHeaderInfo ( CSVFormat . Header . None )
                                . withRecordDelimiter ( "\n" )
                                . withFieldDelimiter ( "|" ) )
                                . withSplitRange ( start , end )
                                . withOutputSerialization ( new
                                    OutputSerialization ()
                                        . withCsvOutputFormat ( new CSVFormat ()
                                            . withCrcEnabled ( true ) ) ) ) ) ;

            selectObjectRequest . setExpression ( sql ) ;
            OSSObject ossObject = client . selectObject (
                selectObjectRequest ) ;
            byte [] buffer = new byte [ 4096 ] ;
            int bytesRead ;
            int totalSize = 0 ;
            try {
                while ( ( bytesRead = ossObject . getObjectContent ()
                    . read ( buffer ) ) != - 1 ) {
                    totalSize += bytesRead ;
                }
                String result = new String ( buffer , 0 ,
                    totalSize - 1 ) ;
                return new Integer ( Integer . parseInt ( result ) ) ;
            } catch ( IOException e ){
                System . out . println ( e . toString () ) ;
                return new Integer ( 0 ) ;
            }
        }
    }

    class RoleCredentialProvider {
        public static final String REGION_CN_HANGZHOU = "cn -
hangzhou " ;
        // Obtain the current Security Token Service ( STS )
        API version .
        public static final String STS_API_VERSION = " 2015 -
04 - 01 " ;
    }

```

```

    public static final String serviceAccessKeyId = "<
AccessKey ID that can play the assumed role >";
    public static final String serviceAccessKeySecret =
"< AccessKey Secret >";

    public static final long DurationSeconds = 15 * 60
;

    private Credentials credential;
    private Calendar expireTime;

    private String roleArn;
    private DefaultAcsClient client;

    public RoleCredentialProvider ( String roleArn ) throws
InvalidCredentialsException {
        this . roleArn = roleArn ;
    }

    private AssumeRoleResponse assumeRole ( String
accessKeyId , String accessKeySecret , String roleArn ,
String roleSessionName , String policy , ProtocolType
protocolType , long durationSeconds ) throws ClientExce
ption {
        try {
            // Create an AcsClient instance for sending
API requests .
            if ( this . client == null ) {
                IClientProfile profile = DefaultProfile .
getProfile ( REGION_CN_HANGZHOU , accessKeyId , accessKeyS
ecret );
                this . client = new DefaultAcsClient ( profile
);
            }
            // Create an AssumeRole Request instance and
set its properties .
            final AssumeRoleRequest request = new
AssumeRoleRequest ();
            request . setVersion ( STS_API_VERSION );
            request . setMethod ( MethodType . POST );
            request . setProtocol ( protocolType );
            request . setRoleArn ( roleArn );
            request . setRoleSessionName ( roleSessionName );
            request . setPolicy ( policy );
            request . setDurationSeconds ( durationSeconds );
            // Send the request and obtain the response .
            final AssumeRoleResponse response = client .
getAcsResponse ( request );
            return response ;
        } catch ( ClientException e ) {
            throw e ;
        }
    }

    public CredentialsProvider GetCredentialProvider ()
throws IOException {
        // Request parameters for the AssumeRole API
include RoleArn , RoleSessionName , Policy , and DurationSe
conds .
        // You need to obtain the value of the
RoleArn parameter in the Resource Access Management (
RAM ) console .

```



```

        // The RoleSessionName parameter indicates the
        name of the session for the temporary token. You
        can use this parameter to identify users in audit
        or identify users who you want to issue tokens to.
        // However, you need to pay attention to the
        length and rules of the RoleSessionName parameter. It
        can contain only letters, numbers, hyphens (-), and
        underscores (_), and cannot include spaces.
        // For more information about the rules, see
        the format requirements in the API reference.
        SimpleDateFormat timeFormat = new SimpleDateFormat
        (" yyyy - MM - dd ");
        String roleSessionName = " AssumingRole " +
        timeFormat.format( Calendar.getInstance().getTime());
        // Read OSS data.
        String policy = "{\n" +
            "    \"Version\": \"1\", \n" +
            "    \"Statement\": [\n" +
            "        {\n" +
            "            \"Action\": \"oss:*\", \n" +
            "            \"Resource\": [\n" +
            "                \"acs:oss:*:*:*\" \n" +
            "            ], \n" +
            "            \"Effect\": \"Allow\" \n" +
            "        }\n" +
            "    ]\n" +
            "}";
        // You must set the protocol type to HTTPS.
        ProtocolType protocolType = ProtocolType.HTTPS;
        try {
            final AssumeRoleResponse response = assumeRole(
                serviceAccessKeyId, serviceAccessKeySecret,
                roleArn, roleSessionName, policy,
                protocolType, DurationSeconds);
            String ossAccessId = response.getCredentials
            ().getAccessKeyId();
            String ossAccessKey = response.getCredentials
            ().getAccessKeySecret();
            String ossSts = response.getCredentials().
            getSecurityToken();

            return new DefaultCredentialProvider(
                ossAccessId, ossAccessKey, ossSts);

        } catch (ClientException e) {
            throw new InvalidCredentialsException(" Unable
            to get the temporary AK:" + e);
        }

        public void setClient( DefaultAcsClient client ) {
            this.client = client;
        }

        public void setCredentials( Credentials creds ) {
            this.credential = creds;
        }

        public Credentials getCredentials() {
            if ( credential != null && expireTime.after(
                Calendar.getInstance())) {
                return credential;
            }

```

```

        try {
            Credential sProvider provider = GetCredent
            ialProvide r ();
            credential = provider . getCredent ials ();
            expireTime = Calendar . getInstanc e ();
            expireTime . add ( Calendar . SECOND , ( int )
            DurationSe conds - 60 );
            return credential ;
        } catch ( IOExceptio n e ) {
            throw new InvalidCre dentialsEx ception (" Unable
            tp get the temporary AK : " + e );
        }
    }
}

public class SelectObje ctSample {
    private static String endpoint = "< OSS endpoint >";
    private static String bucketName = "< Bucket >";
    private static String key = "< Object >";
    private static String roleArn = "< Service role ' s
    ARN >"; // You can obtain the Alibaba Cloud Resource
    Name ( ARN ) of a RAM role in the RAM console . The
    RAM role must have permission s to access OSS .
    private static String recordDeli miter = "\ n ";
    private static int threadCoun t = 10 ;

    public static void main ( String [] args ) throws
    Exception {
        ClientBuil derConfigu ration config = new ClientBuil
        derConfigu ration ();
        RoleCreden tialProvid er provider = new RoleCreden
        tialProvid er ( roleArn );
        Credential s credential s = provider . getCredent ials
        ();
        // OSS client = new OSSClientB uilder (). build (
        endpoint , accessKeyI d , accessKeyS ecret , config );
        System . out . println ( " Id " + credential s .
        getAccessK eyId ());
        System . out . println ( " Key " + credential s .
        getSecretA ccessKey ());
        System . out . println ( " Token " + credential s .
        getSecurit yToken ());
        OSS client = new OSSClientB uilder (). build (
        endpoint , credential s . getAccessK eyId (), credential s .
        getSecretA ccessKey (), credential s . getSecurit yToken (),
        config );
        int totalSplit s = 1 ;
        try {
            SelectObje ctMetadata selectObje ctMetadata =
            client . createSele ctObjectMe tadata (
            new CreateSele ctObjectMe tadataRequ est (
            bucketName , key )
            . withInputS erializati on (
            new InputSeria lization ( ).
            withCsvInp utFormat (
            new CSVFormat ( ).
            withHeader Info ( CSVFormat . Header . None ). withRecord Delimiter
            ( recordDeli miter ))));

            totalSplit s = selectObje ctMetadata . getCsvObje
            ctMetadata (). getSplits ();
            System . out . println ( selectObje ctMetadata .
            getCsvObje ctMetadata (). getTotalLi nes ());
            System . out . println ( totalSplit s );
        }
    }
}

```

```

    }
    catch ( Exception e )
    {
        e . printStack Trace ();
    }

    String sql = " select count (*) from ossobject ";

    ExecutorService executor = Executors . newFixedTh
readPool ( threadCount );
    long startTime = System . currentTim eMillis ();
    List < Future < Integer >> list = new ArrayList < Future
< Integer >>();
    int n = threadCount < totalSplit s ? threadCount
: totalSplit s ;
    for ( int i = 0 ; i < n ; i ++ ) {
        int start = i * totalSplit s / n ;
        int end = i == n - 1 ? totalSplit s - 1 : ( i
+ 1 ) * totalSplit s / n - 1 ;
        System . out . println ( " start : " + start + " end : "
+ end );
        Callable < Integer > task = new MultipartSe lector (
client , bucketName , key , start , end , sql );
        Future < Integer > future = executor . submit ( task
);
        list . add ( future );
    }

    long totalLines = 0 ;
    for ( Future < Integer > task : list ){
        totalLines += task . get (). longValue ();
    }
    long endTime = System . currentTim eMillis ();
    System . out . println ( " total lines : " + totalLines );
    System . out . printf ( " Total time % dms \ n " , (
endTime - startTime ));
}
}

```

SQL statement examples

• SQL statement examples for CSV objects

Scenario	SQL statement
Returns the first 10 rows.	select * from ossobject limit 10
Returns integers in the first and third columns, in which the integers in the first column are larger than those in the third column.	select _1, _3 from ossobject where cast(_1 as int) > cast(_3 as int)
Returns the number of records in which the data in the first column starts with X. (A Chinese character specified after like must be UTF-8 encoded.)	select count(*) from ossobject where _1 like 'X%'

Scenario	SQL statement
Returns all records in which the time of the data in the second column is later than 2018-08-09 11:30:25 and the data in the third column is larger than 200.	<code>select * from ossobject where _2 > cast('2018-08-09 11:30:25' as timestamp) and _3 > 200</code>
Returns the average value, sum, maximum value, and minimum value of the floating-point numbers in the second column.	<code>select AVG(cast(_2 as double)), SUM(cast(_2 as double)), MAX(cast(_2 as double)), MIN(cast(_2 as double))</code>
Returns all records in which the strings concatenated by the data in the first and third columns start with Tom and end with Anderson.	<code>select * from ossobject where (_1 _3) like 'Tom%Anderson'</code>
Returns all records in which the data in the first column is divisible by 3.	<code>select * from ossobject where (_1 % 3) == 0</code>
Returns all records in which the data in the first column ranges from 1995 to 2012.	<code>select * from ossobject where _1 between 1995 and 2012</code>
Returns all records in which the data in the fifth column is N, M, G, or L.	<code>select * from ossobject where _5 in ('N', 'M', 'G', 'L')</code>
Returns all records in which the product of the data in the second and third columns is larger than the sum of 100 and the data in the fifth column.	<code>select * from ossobject where _2 * _3 > _5 + 100</code>

- SQL statement examples for JSON objects

The following JSON object is used as an example.

```
{
  " contacts ":[
    {
      " firstName ": " John ",
      " lastName ": " Smith ",
      " isAlive ": true ,
      " age ": 27 ,
      " address ": {
        " streetAddress ": " 21 2nd Street ",
        " city ": " New York ",
        " state ": " NY ",
        " postalCode ": " 10021 - 3100 "
      },
      " phoneNumbers ": [
        {
          " type ": " home ",
          " number ": " 212 555 - 1234 "
        },
        {


```

```

    " type ": " office ",
    " number ": " 646    555 - 4567 "
  },
  {
    " type ": " mobile ",
    " number ": " 123    456 - 7890 "
  }
],
" children ": [],
" spouse ": null
}, ... # Other similar nodes are omitted .
}]

```

The following table describes the SQL statement examples.

Scenario	SQL statement
Returns all records in which the value of age is 27.	select * from ossobject.contacts[*] s where s.age = 27
Returns all home phone numbers.	select s.number from ossobject.contacts[*].phoneNumbers[*] s where s.type = "home"
Returns all records in which the value of spouse is null.	select * from ossobject s where s.spouse is null
Returns all records in which the value of children is left empty.	select * from ossobject s where s.children[0] is null <div>  Note: The preceding statement is used because an empty array cannot be specified in other ways. </div>

Best practices

- Query large objects in multipart queries.

If columns in a CSV object do not include row delimiters, you can divide the object into parts based on bytes. This method is the simplest because you do not need to

create Select Meta for the object. If columns in a CSV object include row delimiters or a JSON object is queried, follow these steps:

1. Call the `create_select_object_meta` operation to obtain the total number of splits for the object. If you need to call the `SelectObject` API for the object, call it asynchronously before the query to shorten the scanning time.
 2. Select the appropriate concurrency `n` based on resources on the client, and divide the total number of splits by the concurrency `n` to get the number of splits to be contained in each query.
 3. Set parameters, such as `split-range=1-20`, in the request body to perform multipart queries.
 4. Merge the results if required.
- Use the `SelectObject` API for normal objects. We recommend that you do not use the `SelectObject` API to query multipart and appendable objects. The differences in their internal structures may deteriorate the query performance.
 - When querying a JSON object, narrow down the JSON path range in the `FROM` statement.

[DO NOT TRANSLATE]

The following JSON object is used as an example.

```
{  contacts :[
    {" firstName ":" John ", " lastName ":" Smith ", "
    phoneNumbers":[{" type ":" home ", " number ":" 212 - 555 - 1234
    "}, {" type ":" office ", " number ":" 646 - 555 - 4567 "}, {" type
    ":" mobile ", " number ":" 123 456 - 7890 "}], " address":{"
    streetAddress ":" 21 2nd Street ", " city ":" New York ",
    " state ":" NY ", " postalCode ":" 10021 - 3100 "}
  ]
}
```

```
  ]}
```

To query all `streetAddress` data of records in which `postalCode` starts with 10021, run the following SQL statement:

```
select s . address . streetAddress from ossobject . contacts  
[*] s where s . address . postalCode like ' 10021 %'
```

Alternatively, run the following SQL statement:

```
select s . streetAddress from ossobject . contacts [*].  
address s where s . postalCode like ' 10021 %'
```

The query performance of the second SQL statement is better because the JSON path is more accurate.

- Process high-accuracy floating-point numbers in JSON objects.

If you need to calculate high-accuracy floating-point numbers in a JSON object, we recommend that you set the `ParseJsonNumberAsString` parameter to `true`, and use the `CAST` function to convert the resolved data into the decimal type. For example, the value of attribute `a` is 123456789.123456789. You can run

```
select s . a  
from ossobject s where cast ( s . a as decimal ) >  
123456789 . 12345 to maintain the accuracy of attribute a.
```

11.7 Object tagging

You can use the object tagging function to classify OSS objects. You can set lifecycle rules for objects with the same tag in batches.



Note:

The object tagging function is in the beta testing phase. You can [open a ticket](#) to apply for the trial of this function.

The object tagging function uses a key-value pair to tag an object. You can add a tag to an object when uploading the object or add a tag to an existing object.

- You can add up to 10 tags with different keys to an object.
- The maximum length of a key and a value is 128 bytes and 256 bytes individually.
- The key and value are both case-sensitive.
- A tag can contain letters, numbers, spaces, and the following symbols: + - = . _ : /

- Only the owner of a bucket and authorized users can read and write the tags of an object in the bucket. The read and write permissions on the tags of an object are not restricted by the ACL for the object.
- When you copy an object to another region, the tags of the object is also copied to the region.

Application scenarios

The object tagging function is not limited by folders. You can select all objects with the same tag in a bucket and set lifecycle rules for the objects in batches. For example , you can add tags to objects that are generated periodically and do not need to be stored for a long period, and then set a lifecycle rule for the objects with the tags to delete them periodically.

Usage

- APIs that may be used by the object tagging function are listed as follows:
 - [PutObjectTagging](#): Adds a tag to an object. If the target object already has a tag, the original tag is overwritten by the new tag.
 - [GetObjectTagging](#): Reads the tags of an object.
 - [DeleteObjectTagging](#): Deletes the tags of an object.
 - [PutObject](#): When using PutObject to upload an object, you can specify the `x - oss - tagging` request header to add tags to the object.
 - [InitiateMultipartUpload](#): When initiating a multipart upload task, you can specify the `x - oss - tagging` request header to add tags to the object.
 - [CopyObject](#): When copying an object, you can specify the `x - oss - tagging - directive` request header to determine whether to copy the tags of the original object and specify the `x - oss - tagging` request header to specify the tags of the target object.
 - [GetObject](#): If the user has the permission to read the tags of the target object, the `x - oss - tagging - count` header is included in the response to the GetObject request to indicate the number of tags added to the target object.
 - [HeadObject](#): If the user has the permission to read the tags of the target object, the `x - oss - tagging - count` header is included in the response to the HeadObject request to indicate the number of tags added to the target object.

- Required permission

The permissions required by APIs related to the object tagging function are listed as follows:

- **GetObjectTagging:** Requires the permission to obtain the tags of the object. With this permission, you can view the tags added to an object.
- **PutObjectTagging:** Requires the permission to configure the tags of the object. With this permission, you can configure tags for an object.
- **DeleteObjectTagging:** Requires the permission to delete the tags of the object. With this permission, you can delete the tags of an object.

Object tagging and lifecycle rule management

When setting a lifecycle rule, you can specify the condition of the rule so that it applies to objects with a specified prefix or a specified tag. You can also set the prefix and tag as the condition of a rule at the same time.

- If you set a specified tag as the condition of a lifecycle rule, the rule applies to an object only when the key and value in the tag of the object both match the specified tag.
- If you specify a prefix and multiple tags as the condition of a lifecycle rule, the rule applies to an object only when the prefix and tags of the object match the prefix and all tags specified in the rule.

Example:

```
< LifecycleConfiguration >
  < Rule >
    < ID > r1 </ ID >
    < Prefix > rule1 </ Prefix >
    < Tag >< Key > xx </ Key >< Value > 1 </ Value ></ Tag >
    < Tag >< Key > yy </ Key >< Value > 2 </ Value ></ Tag >
    < Status > Enabled </ Status >
    < Expiration >
      < Days > 30 </ Days >
    </ Expiration >
  </ Rule >
  < Rule >
    < ID > r2 </ ID >
    < Prefix > rule2 </ Prefix >
    < Tag >< Key > xx </ Key >< Value > 1 </ Value ></ Tag >
    < Status > Enabled </ Status >
    < Transition >
      < Days > 60 </ Days >
      < StorageClass > Archive </ StorageClass >
    </ Transition >
  </ Rule >
```

```
</ LifecycleConfiguration >
```

If you set the preceding lifecycle rule:

- Objects with the rule1 prefix and "xx=1" and "yy=2" tags are deleted after 30 days.
- The storage class of objects with the rule2 prefix and "xx=1" tag is converted to Archive after 60 days.



Note:

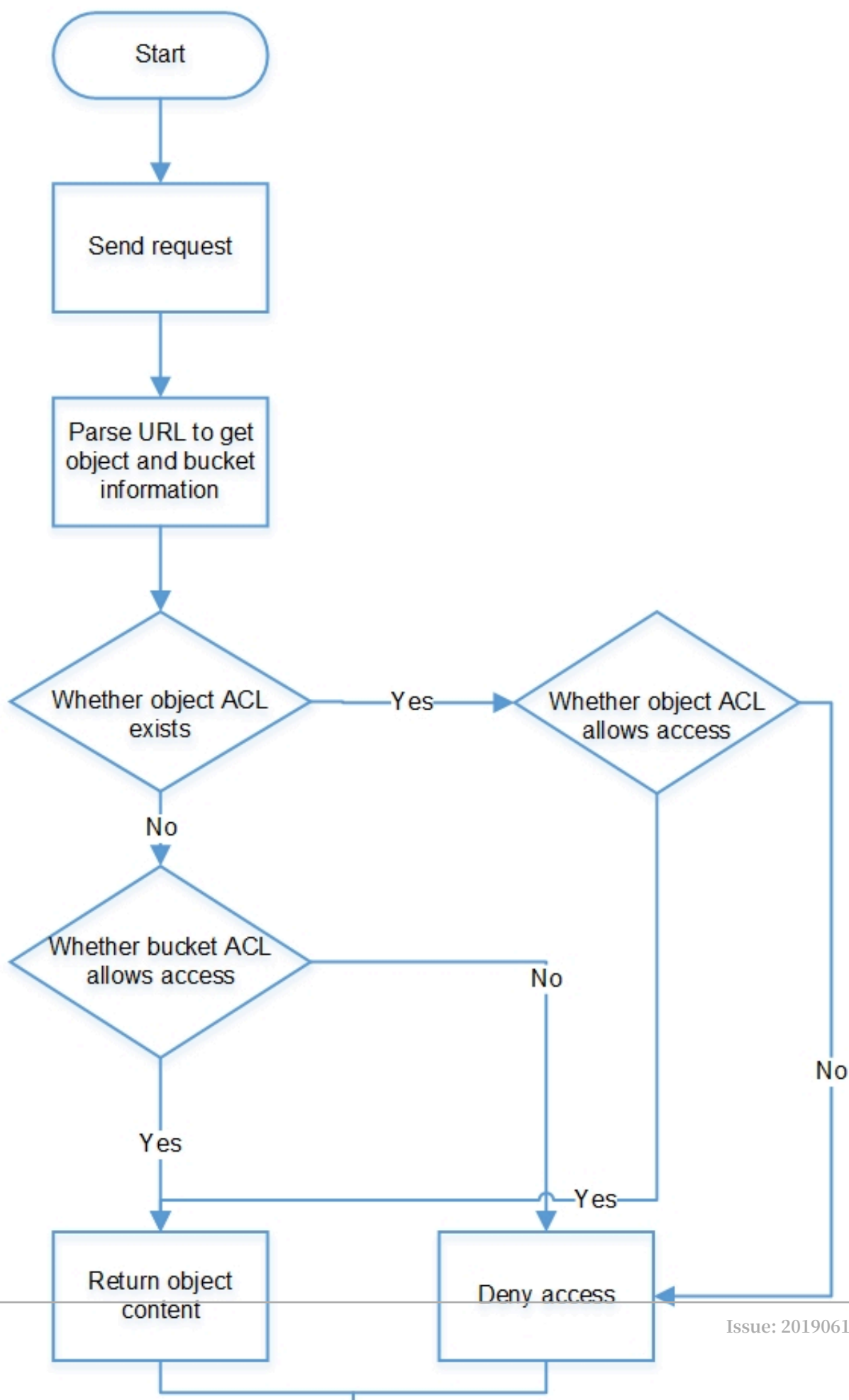
For more information, see [Manage lifecycle rules](#).

12 Signature

12.1 OSS request process

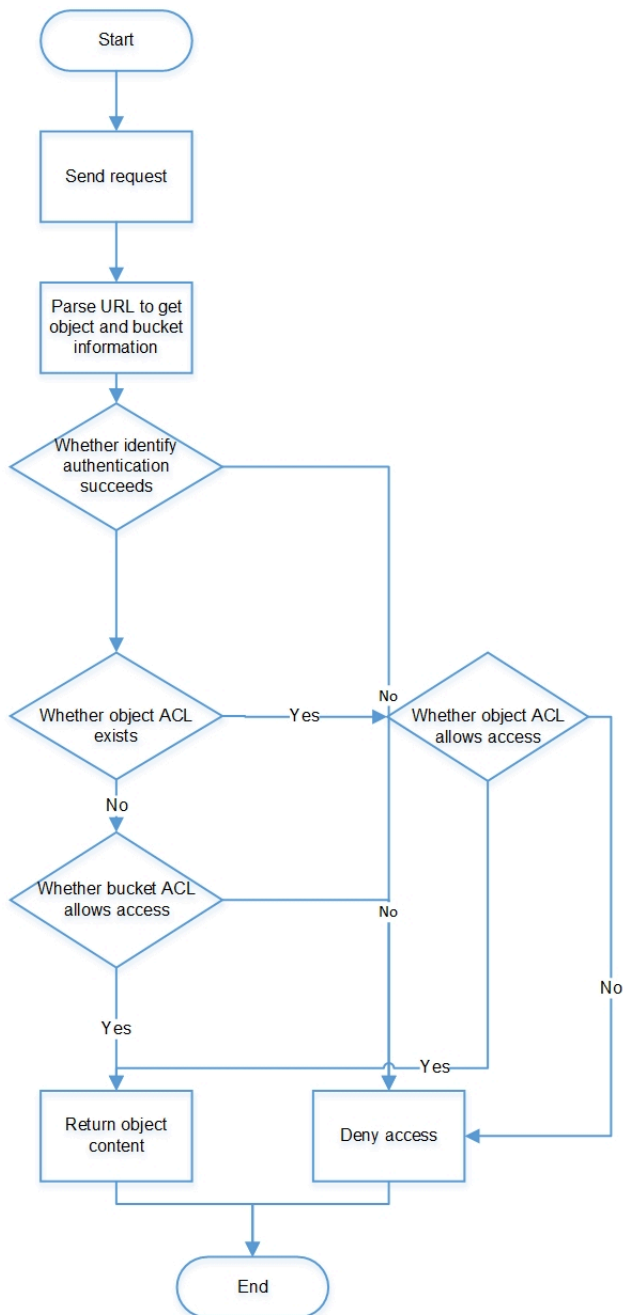
Based on whether the authentication information is included, HTTP requests sent to OSS are divided into two types: requests with authentication information and anonymous requests without authentication information. An anonymous request does not include authentication information. In contrast, a request with authentication information includes signature information in the request header or request URL, which is in compliance with the OSS API documents.

Access to OSS using anonymous requests



1. A user request is sent to the HTTP server of OSS.
2. OSS parses the URL to obtain the target bucket and object.
3. OSS checks whether an ACL is set for the object.
 - If no ACL is set for the object, the process proceeds to step 4.
 - If an ACL is set for the object, OSS checks whether the ACL allows anonymous access.
 - If the ACL allows anonymous access, the process proceeds to step 5.
 - If the ACL does not allow anonymous access, the request is rejected and the process ends.
4. OSS checks whether the bucket ACL allows anonymous access.
 - If the ACL allows anonymous access, the process proceeds to step 5.
 - If the ACL does not allow anonymous access, the request is rejected and the process ends.
5. The request passes the authentication, and the object content is returned to the user.

Access to OSS using requests with authentication information



1. A user request is sent to the HTTP server of OSS.
2. OSS parses the URL to obtain the target bucket and object.

3. OSS obtains the identity information about the requester for authentication based on the AccessKeyId of the request.
 - If the identity information is not obtained, the request is rejected and the process ends.
 - If the identity information is obtained, but the requester is not allowed to access the resource, the request is rejected and the process ends.
 - If the identity information is obtained, but the signature calculated based on the HTTP parameters in the request does not match the signature contained in the request, the request is rejected and the process ends.
 - If the authentication succeeds, the process proceeds to step 4.
4. OSS checks whether an ACL is set for the object.
 - If no ACL is set for the object, the process proceeds to step 5.
 - If an ACL is set for the object, OSS checks whether the object ACL allows access by the user.
 - If the ACL allows access by the user, the process proceeds to step 6.
 - If the ACL does not allow the access, the request is rejected and the process ends.
5. OSS checks whether the bucket ACL allows access by the user.
 - If the ACL allows access by the user, the process proceeds to step 6.
 - If the ACL does not allow access by the user, the request is rejected and the process ends.
6. The request passes the authentication, and the object content is returned to the user.

AccessKey types

Currently, the following three types of [AccessKeys](#) (AKs) are used to access OSS:

- AK of an Alibaba Cloud account

An AK of an Alibaba Cloud account indicates the AK of the bucket owner. The AK of an Alibaba Cloud has full access to all resources under the corresponding account. Each Alibaba Cloud account can have a maximum of five AK pairs

(AccessKeyId and AccessKeySecret) and the AKs can be in either an active or inactive state.

You can log on to the [AccessKey console](#) to add or delete AK pairs.

An AK pair can be in two states: active and inactive.

- An AK in the active state can be used for authentication.
- An AK in the inactive state cannot be used for authentication.



Notice:

For security reasons, avoid using the AK of your Alibaba Cloud account.

- **AK of a RAM user**

Resource Access Management (RAM) is a resource access control service provided by Alibaba Cloud. AKs of RAM users are authorized by the corresponding Alibaba Cloud account through RAM. These AKs can be used only to access OSS resources in buckets in accordance with the rules defined in RAM. By configuring RAM policies, you can manage multiple users in a centralized manner and control the resources that can be accessed by the users. For example, you can control the permission of a user so that the user can only read a specified bucket. A RAM user is subjected to the Alibaba Cloud account under which it was created, and does not own any actual resources. That is, all resources belong to the corresponding Alibaba Cloud account.

- **AK of an STS account**

Security Token Service (STS) is an Alibaba Cloud service that provides temporary access credentials. AKs of STS accounts are authorized by STS. These AKs can be used only to access OSS resources in buckets in accordance with the rules defined in STS.

Authentication implementation

Authentication is implemented in the following three methods:

- **AK authentication**
- **RAM authentication**
- **STS authentication**

When a user sends a request to OSS as an individual identity, authentication is performed on the user as follows:

1. The user generates a signature string based on the request in the format specified by OSS.
2. The user uses the AccessKeySecret to encrypt the signature string and generate a verification code.
3. After receiving the request, OSS locates the corresponding AccessKeySecret based on the AccessKeyId, and obtains the signature string and verification code using the same method.
 - If the calculated verification code is the same as the provided verification code, OSS determines that the request is valid.
 - If the obtained verification code is different from the provided verification code, OSS rejects the request and returns an HTTP 403 error.

Three methods of accessing OSS with authentication

- Access OSS in the console: The authentication process is invisible to users, which means users do not need to worry about authentication configurations when they access OSS in the console. For more information, see [Download an object](#).
- Access OSS using SDKs: OSS provides SDKs for multiple development languages, in which the signature algorithm is implemented. Therefore, users only need to input the AK information to access OSS using SDKs. For more information, see the access control part in the SDK documents for different development languages, such as [Java SDK: Authorized access](#) and [Python SDK: Authorized access](#).
- Access OSS using APIs: To write code to package a call to the RESTful API, you must implement a signature algorithm to calculate the signature. For more information, see [Add a signature to the header](#) and [Add a signature to a URL](#).

12.2 Add a signature to the header

You can add an authorization header to carry signature information in an HTTP request to indicate that the message has been authorized.

SDK signature implementation

OSS SDK has implemented the signature. You do not need to worry about the signature issue when you use the OSS SDK. To learn more about the signature implementations of specific languages, see the OSS SDK code. The files for implementing OSS SDK signature are shown in the following table:

SDK	Signature implementation
Java SDK	OSSRequestSigner.java
Python SDK	auth.py
Net SDK	OssRequestSigner.cs
PHP SDK	OssClient.php
C SDK	oss_auth.c
JavaScript SDK	client.js
Go SDK	auth.go
Ruby SDK	util.rb
iOS SDK	OSSModel.m
Android SDK	OSSUtils.java

Calculation of the Authorization field

```

Authorization = " OSS " + AccessKeyId + ":" + Signature
Signature = base64 ( hmac - sha1 ( AccessKeySecret ,
    VERB + "\ n "
    + Content - MD5 + "\ n "
    + Content - Type + "\ n "
    + Date + "\ n "
    + CanonicalizedOSSHeaders
    + CanonicalizedResource ))

```

- The `AccessKeySecret` indicates the key required for a signature.
- `VERB` indicates the HTTP request method, including PUT, GET, POST, HEAD, and DELETE.
- `\ n` is a line break.
- `Content - MD5` The Content-MD5 is the MD5 value of requested content data. The message content (excluding the header) is calculated to obtain an MD5 value, which is a 128-bit number. This number is encoded with Base64 into a Content-MD5 value. The request header can be used to check the message validity, that is, whether the message content is consistent with the sent content, such as “eB5eJF1ptWaXm4bijSPyxw==”. The request header may be empty. For more information, see [RFC2616 Content-MD5](#).
- `Content - Type` indicates the requested content type, such as “application/octet-stream”. It content type may be empty.

- `Date` indicates the time that the operation takes. It must be in GMT format, such as “Sun, 22 Nov 2015 08:16:38 GMT” .
- The `CanonicalizedOSSHeaders` indicates an assembly of HTTP headers whose prefixes are “x-oss-” .
- The `CanonicalizedResource` indicates the OSS resource that the user wants to access.

Specifically, the values of `Date` and `CanonicalizedResource` cannot be empty. If the difference between the value of `Date` in the request and the time of the OSS server is greater than 15 minutes, the OSS server rejects the request and returns an HTTP 403 error.

Construct CanonicalizedOSSHeaders

All the HTTP headers whose prefixes are x-oss- are called CanonicalizedOSSHeaders. The method to construct CanonicalizedResource is as follows:

1. Convert the names of all HTTP request headers whose prefixes are x-oss- into lowercase letters. For example, convert `X - OSS - Meta - Name : TaoBao` to `x - oss - meta - name : TaoBao` .
2. If the request is sent with the `AccessKeyID` and `AccessKeySecret` obtained by the STS, you must also add the obtained security-token value to the signature string in the form of `x - oss - security - token : security - token` .
3. Sort all acquired HTTP request headers in a lexicographically ascending order.
4. Delete any space on either side of a separator between the request header and content. For example, convert `x - oss - meta - name : TaoBao` to `x - oss - meta - name : TaoBao` .
5. Separate all the content and headers with the `\ n` separator to form the final CanonicalizedOSSHeaders.



Note:

- CanonicalizedOSSHeaders can be empty, and the `\ n` at the end can be removed.
- If only one header must be constructed, it must be `x - oss - meta - a \ n` .
Note the `\ n` at the end.

- If multiple headers must be constructed, it must be `x - oss - meta - a : a \n x - oss - meta - b : b \n x - oss - meta - c : c \n`. Note the `\n` at the end.

Construct CanonicalizedResource

The target OSS resource specified in the request sent by the user is called a CanonicalizedResource. The method for constructing CanonicalizedResource is as follows:

1. Set CanonicalizedResource into a null character string (`"`);
2. Add the OSS resource to be accessed in the following format: `/ BucketName / ObjectName` . (If ObjectName does not exist, CanonicalizedResource is `" / BucketName / "`. If BucketName does not exist either, CanonicalizedResource is `" / "`.)
3. If the requested resource includes sub-resources (SubResource), sort all the sub-resources in a lexicographically ascending order and separate the sub-resources using the separator `&` to generate a sub-resource string. Add `"?"` and the sub-resource string to the end of the CanonicalizedResource string. In this case, CanonicalizedResource is like: `/ BucketName / ObjectName ? acl & uploadId = UploadId`



Note:

- The sub-resources supported by OSS currently include: `acl`, `uploads`, `location`, `cors`, `logging`, `website`, `referer`, `lifecycle`, `delete`, `append`, `tagging`, `objectMeta`, `uploadId`, `partNumber`, `security-token`, `position`, `img`, `style`, `styleName`, `replication`, `replicationProgress`, `replicationLocation`, `cname`, `bucketInfo`, `comp`, `qos`, `live`, `status`, `vod`, `startTime`, `endTime`, `symlink`, `x-oss-process`, `response-content-type`, `response-content-language`, `response-expires`, `response-cache-control`, `response-content-disposition`, and `response-content-encoding`.
- Three types of sub-resources are available:
 - Resource identifiers, such as `acl`, `append`, `uploadId`, and `symlink` sub-resources. For more information, see [Bucket-related operations](#) and [Object-related operations](#).
 - Specify response header fields such as `response -***`. For more information, see the `Request Parameters` section of [GetObject](#) .
 - Object handling methods, such as `x - oss - process` . It is used as the object handling method, such as [Image Processing](#).

Rules to calculate a signature header

- A signature string must be in the UTF-8 format. Encode a signature string containing Chinese characters with UTF-8 first, and then use it with the AccessKeySecret to calculate the final signature.
- The signing method adopted is the HMAC-SHA1 method defined in [RFC 2104](#), where Key is `AccessKeySecret`.
- Content-Type and Content-MD5 are not required in a request. If the request requires signature verification, the null value can be replaced with the line break “\n”.
- Among all non-HTTP-standard headers, only the headers starting with “x-oss-” require signature strings, and other non-HTTP-standard headers are ignored by OSS. (For example, the “x-oss-magic” header in the preceding example must be added with a signature string.)
- Headers starting with “x-oss-” must comply with the following specifications before being used for signature verification:
 - The header name is changed to lower-case letters.
 - The headers are sorted in a lexicographically ascending order.
 - No space exists before and after the colon, which separates the header name and value.
 - Each header is followed by the line break “\n”. If no header is used, CanonicalizedOSSHeaders is set to null.

Example signature

Assume that AccessKeyID is 44CF9590006BF252F707 and AccessKeySecret is OtxrxzIsfpFjA7SwPzILwy8Bw21TLhquhboDYROV.

Request	Signature string calculation formula	Signature string
PUT /nelson HTTP/1.0 Content-MD5: eB5eJF1ptW aXm4bijSPyxw== Content- Type: text/html Date: Thu, 17 Nov 2005 18:49:58 GMT Host: oss-example.oss-cn- hangzhou.aliyuncs.com X-OSS-Meta-Author: foo @bar.com X-OSS-Magic: abracadabra	Signature = base64(hmac-sha1(AccessKeyS ecret,VERB + “\n” + Content-MD5 + “\n ” + Content-Type + “\ n” + Date + “\n” + CanonicalizedOSSHeaders + CanonicalizedResource))	“PUT\n eB5eJF1ptW aXm4bijSPyxw==\n text/ html\n Thu, 17 Nov 2005 18 :49:58 GMT\n x-oss-magic: abracadabra\n x-oss-meta- author:foo@bar.com\n/oss -example/nels

The signature calculation method is as follows:

Python sample code:

```
import base64
import hmac
import sha
h = hmac.new("0txrxIsfp FjA7SwPzIL wy8Bw21TLh quhboDYROV ",
             "PUT \n0DBG0ERFM DMzQTczRUY 3NUE3NzA5Q zdfNUYzMDQ\n"
             "text / html \nThu , 17 Nov 2005 18 : 49 : 58 GMT\n"
             "x-oss-magic : abracadabra \n x-oss-meta-author : foo @\n"
             "bar . com \n / oss - example / nelson ", sha)
Signature = base64.b64encode(h.digest())
print("Signature : %s" % Signature)
```

The signature calculation result is 26NBxoKdsyly4EDv6inkoDft/yA=. According to the formula Authorization = “OSS “ + AccessKeyID + “:” + Signature, the value of Authorization is OSS 44CF9590006BF252F707:26NBxoKdsyly4EDv6inkoDft/yA=. The value is added with the authorization header to form the message to be sent:

```
PUT /nelson HTTP / 1 . 0
Authorizat ion : OSS 44CF959000 6BF252F707 : 26NBxoKdsy
ly4EDv6ink oDft / yA =
Content - Md5 : eB5eJF1ptW aXm4bijSPy xw ==
Content - Type : text / html
Date : Thu , 17 Nov 2005 18 : 49 : 58 GMT
Host : oss - example . oss - cn - hangzhou . aliyuncs . com
X - OSS - Meta - Author : foo @ bar . com
X - OSS - Magic : abracadabr a
```

Detail analysis are as follows:

- If the input AccessKeyID does not exist or is inactive, the error 403 Forbidden is returned. Error code: InvalidAccessKeyId.

- If the authorization value format in the user request header is incorrect, the error 400 Bad Request is returned. Error code: InvalidArgument.
- All the requests of OSS must use the GMT time format stipulated by the HTTP 1.1 protocol. Specifically, the date format is: `date1 = 2DIGIT SP month SP 4DIGIT ; day month year (for example , 02 Jun 1982)`. In the aforesaid date format, “day” occupies “2 digits”. Therefore, “Jun 2”, “2 Jun 1982”, and “2-Jun-82” are all invalid date formats.
- If Date is not input into the header or the format is incorrect during signature verification, the error 403 Forbidden is returned. Error code: AccessDenied.
- The request must be entered within 15 minutes based on the current time of the OSS server; otherwise, the error 403 Forbidden is returned. Error code: RequestTimeTooSkewed.
- If the AccessKeyID is active but OSS determines that the signature of the user request is incorrect, the error 403 Forbidden is returned, and the correct signature string for verification and encryption is returned to the user in the response message. The user can check whether or not the signature string is correct based on the response of OSS. Return example:

```
<? xml version = " 1 . 0 " ? >
< Error >
  < Code >
    SignatureDoesNotMatch
  </ Code >
  < Message >
    The request signature we calculated does not
    match the signature you provided . Check your key
    and signing method .
  </ Message >
  < StringToSignBytes >
    47 45 54 0a 0a 0a 57 65 64 2c 20 31 31
    20 4d 61 79 20 32 30 31 31 20 30 37 3a
    35 39 3a 32 35 20 47 4d 54 0a 2f 75 73 72
    65 61 6c 74 65 73 74 3f 61 63 6c
  </ StringToSignBytes >
  < RequestId >
    1E446260FF 9B10C2
  </ RequestId >
  < HostId >
    oss - cn - hangzhou . aliyuncs . com
  </ HostId >
  < SignatureProvided >
    y5H7yzPsA / tP4 + 0tH1HHvPEw Uv8 =
  </ SignatureProvided >
  < StringToSign >
    GET
    Wed , 11 May 2011 07 : 59 : 25 GMT
    / oss - example ? acl
  </ StringToSign >
  < OSSAccessKeyId >
```

```

    AKIAIVAKMS  MOY7VOMRWQ
  </ OSSAccessK  eyId >
</ Error >

```

Content-MD5 calculation method

```

Content - MD5 calculation
The message content "123456789" is used as an example.
The Content - MD5 value of the string is calculated as follows:
The algorithm defined in related standards can be simplified to the following:
Calculate the MD5 - encrypted 128 - bit binary array.
Encode the binary array (instead of the 32 - bit string code) with Base64.
Python is used as an example.
The correct calculation code is:
>>> import base64, hashlib
>>> hash = hashlib.md5()
>>> hash.update("0123456789")
>>> base64.b64encode(hash.digest())
'eB5eJF1ptW aXm4bijSPy xw =='
Note:
The correct code is: hash.digest(), used to calculate a 128 - bit binary array
>>> hash.digest()
'x \ x1e ^$] i \ xb5f \ x97 \ x9b \ x86 \ xe2 \ x8d #\ xf2 \ xc7 '
The common error is to base 64 the computed 32 - Bit String encoding directly.
An incorrect example: hash.hexdigest(), and a visible 32 - bit string is calculated.
>>> hash.hexdigest()
'781e5e245d 69b566979b 86e28d23f2 c7 '
Result of encoding the incorrect MD5 value with Base64:
>>> base64.b64encode(hash.hexdigest())
'NzgxZTVlMj Q1ZDY5YjU2 Njk3OWI4Nm UyOGQyM2Yy Yzc ='

```

12.3 Add a signature to a URL

In addition to using an authorization header, you can add signature information to a URL. It enables you to forward a URL to the third party for an authorized access.

Sample code

Python sample code used to add a signature to a URL:

```

import base64
import hmac
import sha
import urllib
h = hmac.new("OtxrzxIsfp FjA7SwPzIL wy8Bw21TLh quhboDYROV ",
             "GET \n \n \n114188912 0 \n / oss - example / oss - api . pdf ",
             sha)

```



```
urllib . quote ( base64 . encodestri ng ( h . digest ( ) ) . strip ( ) )
```

OSS SDK provides the method for adding a signature into an URL. For the detailed usage, see Authorized access in the OSS SDK Reference.

To add a signature to the OSS SDK URL, see the following table.

SDK	URL signature method	Implementation file
Java SDK	OSSClient.generatePresignedUrl	OSSClient.java
Python SDK	Bucket.sign_url	api.py
Net SDK	OssClient.GeneratePresignedUri	OssClient.cs
PHP SDK	OssClient.signUrl	OssClient.php
JavaScript SDK	signatureUrl	object.js
C SDK	oss_gen_signed_url	oss_object.c

Implementation

URL signature example:

```
http :// oss - example . oss - cn - hangzhou . aliyuncs . com / oss - api . pdf ? OSSAccessK eyId = nz2pc56s93 6 ** 9l & Expires = 1141889120 & Signature = vjbyPxybdZ aNmGa % 2ByT272YEA iv4 % 3D
```

The URL signature must include at least the following three parameters: Signature, Expires, and OSSAccessK eyId.

- The Expires parameter indicates the time-out period of a URL. The value of this parameter is UNIX time (which is the number of seconds that have elapsed since 00:00:00 UTC, January 1, 1970. For more information, see [Wikipedia](#)). If the time when OSS receives the URL request is later than the value of the Expires parameter and is included in the signature, an error code request timed-out is returned. For example, if the current time is 1141889060, to create a URL that is scheduled to expire in 60 seconds, you can set the value of Expires to 1141889120. The valid period of a URL is 3,600 seconds by default and 64,800 seconds in maximum.
- OSSAccessK eyId refers to the AccessKeyID in the key.

- **Signature** indicates the signature information. For all requests and header parameters that OSS supports, the algorithm for adding a signature to a URL is basically the same as that of [Adding a signature to a header](#).

```
Signature = urlencode ( base64 ( hmac - sha1 ( AccessKeyS  ecret
',
    VERB + "\ n "
+ CONTENT - MD5 + "\ n "
+ CONTENT - TYPE + "\ n "
+ EXPIRES + "\ n "
+ Canonicali zedOSSHead ers
+ Canonicali zedResourc e )))
```

The difference is listed as follows:

- When a signature is added to a URL, the Expires parameter replaces the Date parameter.
- Signatures cannot be included in a URL and the Header at the same time.
- If more than one incoming Signature, Expires, or AccessKeyId value is available, the first of each incoming value is used.
- Whether the request time is later than the Expires time, is verified first before verifying the signature.
- When you put the signature string into a URL, remember to perform the `UrlEncode` for a URL.
- When you add a signature to a temporary user URL, the `security - token` must also be entered. The format is as follows:

```
http :// oss - example . oss - cn - hangzhou . aliyuncs . com / oss
- api . pdf ? OSSAccessK eyId = nz2pc56s93 6 ** 9l & Expires =
1141889120 & Signature = vjbyPxybdZ aNmGa % 2ByT272YEA iv4 % 3D &
security - token = SecurityTo ken
```

Detail analysis

- If you adopt the approach of adding a signature to a URL, the authorized data is exposed on the Internet before the authorization period expires. We recommend that you must assess the usage risks in advance.
- The PUT and GET requests both support adding a signature in a URL.
- When a signature is added to a URL, the sequence of Signature, Expires, and AccessKeyId can be swapped. If one or more Signature, Expires, or AccessKeyId parameter is missing, the error 403 Forbidden is returned. Error code: AccessDenied.

- If the current access time is later than the Expires time set in the request, the error 403 Forbidden is returned. Error code: AccessDenied.
- If the format of the Expires time is incorrect, the error 403 Forbidden is returned. Error code: AccessDenied.
- If the URL includes one or more Signature, Expires, or AccessKeyId parameter and the header also includes signature information, the error 400 Bad Request is returned. Error code: InvalidArgument.
- When the signature string is generated, the Date parameter is replaced by the Expires parameter, but the headers such as content-type and content-md5 defined in the preceding section are still included. (Though the Date request header still exists in the request, you can skip adding it to the signature string.)

13 Identity authentication

13.1 What is RAM and STS

RAM and STS are permission management systems provided by Alibaba Cloud.

RAM is primarily used to control account system permissions. RAM enables users to create subaccounts within the range of primary account permissions. Different subaccounts can be allocated different permissions for authorization management.

STS is a security credential (token) management system that grants temporary access permissions. STS allows users to grant access rights to the temporary accounts.

Why RAM and STS?

RAM and STS are designed to resolve the core issue such as how to securely grant access permissions to other users without disclosing the primary account's AccessKey. Disclosure of AccessKey poses a serious security threat because unauthorized users may operate account resources and the risk of data leakage or stealing of important information is high.

RAM provides a long-term permission control mechanism. Various subaccounts assign different permissions to the different users. This way, even the disclosure of subaccount information would not cause a global information leakage. However, subaccounts have long-term validity.



Note:

Therefore, AccessKey of subaccounts must not be disclosed.

On the contrary, STS provides temporary access authorization by returning a temporary AccessKey and the token. This information can be provided directly to the temporary accounts, allowing them access to OSS. Generally, the permissions obtained from STS are more restrictive and only valid for a limited period of time. Thus, the disclosure of this information has little effect on the system.

These functions are further illustrated with the help of examples.

Basic concepts

The following are some explanations of the basic concepts:

- **Subaccount:** A subaccount is created from the Alibaba Cloud primary accounts . Once created, it is assigned an independent password and permissions. Each subaccount has its own AccessKey and can perform authorized operations similar to the primary account. Generally, subaccounts can be understood as users with certain permissions or operators with permissions to perform specific operations.
- **Role:** Role is a virtual concept for certain operation permissions. However, it does not have independent logon passwords or AccessKeys.

**Note:**

Subaccounts can assume roles. When a role is assumed, the permissions granted for a subaccount are the permissions of the role.

- **Policy:** Policies are rules used to define permissions; for example, they permit users to read or write certain resources.
- **Resource:** Resources are the cloud resources that users can access like all OSS buckets, a certain OSS bucket, or a certain object in a specific OSS bucket.

A subaccount and roles have the same relationship to each other as you and your identities. At work, you may be an employee, while at home you may be a father. In different scenarios, you may assume different roles. Different roles are assigned corresponding permissions. The concept of “employee” or “father” is not an actual entity that can be the subject of actions. These concepts are only complete when an individual assumes them. This illustrates an important concept: a role may be assumed by multiple people at the same time.

**Note:**

Once the role is assumed, this individual automatically obtains all the permissions of the role.

The following example provides better understanding of the concept:

- Assume that Alice is the the Alibaba Cloud user and she has two private OSS buckets, `alice_a` and `alice_b`. Alice has full permission for both buckets.
- To avoid leaking her Alibaba Cloud account AccessKey, which would pose a major security risk, Alice uses RAM to create two subaccounts, Bob and Carol. Bob has read/write permission for `alice_a` and Carol has read/write permission for `alice_b` . Bob and Carol both have their own AccessKeys. This way, if one is leaked, only

the corresponding bucket is affected and Alice can easily cancel the leaked user permissions on the console.

- Now, for some reason, Alice must authorize another person to read the objects in `alice_a`. In this situation, she must not only disclose Bob's `AccessKey`. Rather, she can create a new role like `AliceAReader`, and grant this role the read permission for `alice_a`. However, note that, at this time, `AliceAReader` cannot be used because no `AccessKey` corresponds to this role. `AliceAReader` is currently only a virtual entity with the permission to access `alice_a`.
- To obtain temporary authorization, Alice can call the STS' `AssumeRole` interface to notify STS that Bob wants to assume the `AliceAReader` role. If successful, STS returns a temporary `AccessKeyId`, `AccessKeySecret`, and `SecurityToken`, which serve as the access credentials. When these credentials are given to a temporary account, the user obtains temporary permission to access `alice_a`. The credentials' expiration time is specified when the `AssumeRole` interface is called.

Why are RAM and STS so complex?

Initially, RAM and STS concepts seem to be complex. This is because flexibility is given to permission control at the cost of simplicity.

Subaccounts and roles are separated to separate the entity that executes operations from the virtual entity that represents a permissions set. If a user requires many permissions including the read and write permissions but each operation only requires part of the total permission set, you can create two roles, one with the read permission and the other with the write permission. Then create a user who does not have any permission but can assume these two roles. When the user needs to read or write data, the user can temporarily assume the role with the read permission or the role with the write permission. This reduces the risk of permission leaks for each operation. Additionally, roles can be used to grant permissions to other Alibaba Cloud users, making the collaboration easier.

Here, flexibility does not mean you have to use all these functions. You only need to use the subset of the functions as required. For example, if you do not need to use temporary access credentials that have an expiration time, you can only use the RAM subaccount function, without STS.

In what follows, we use examples to create a RAM and STS user guide and provide instructions. For the operations in these examples, we do our best to use console and command line operations to reduce the actual amount of codes that must be used. If

you must use code to perform these operations, we recommend that you see the RAM and STS API Manual.

Test tool

During testing, we use `osscli`, a tool in the OSS PythonSDK that allows you to directly work on OSS through the command line. `osscli` can be obtained from [PythonSDK](#).

Typical `osscli` usage:

```
Download files
./ osscli get oss :// BUCKET / OBJECT LOCALFILE -- host =
Endpoint -i AccessKeyId -k AccessKeySecret
Here, replace BUCKET and OBJECT with your own bucket
and object, and the endpoint format must be similar
to oss-cn-hangzhou.aliyuncs.com. For AccessKeyId
and AccessKeySecret, use the information correspond
ing to your own account
Upload files
./ osscli put LOCALFILE oss :// BUCKET / OBJECT -- host =
Endpoint -i AccessKeyId -k AccessKeySecret
The meaning of each field is the same as for the
download example
```

13.2 RAM user

With Alibaba Cloud RAM, you can create RAM users under your Alibaba Cloud account. Each RAM user has their own AccessKeys. In this case, your Alibaba Cloud account is referred to as the primary account and the created RAM users are referred to as the sub-accounts. The AccessKey of a sub-account can be used only to perform operations authorized by your Alibaba Cloud account and use resources authorized by your Alibaba Cloud account.

Scenario

If multiple users need to use resources under your Alibaba Cloud account, they can only use the AccessKey of your Alibaba Cloud account to access the resources. If this occurs, the following two issues arise:

- Your AccessKey is exposed to multiple users, which increases the risk of mistakenly exposing its contents.
- You cannot control which user or users can access specific resources (such as buckets).

To resolve the preceding issues, you can use Alibaba Cloud RAM to create RAM users with their own AccessKeys under your Alibaba Cloud account. In this case, your

Alibaba Cloud account is referred to as the primary account and the created RAM users are referred to as the sub-accounts. You can perform only operations The AccessKey of a sub-account can be used only to perform operations authorized by your Alibaba Cloud account and use resources authorized by your Alibaba Cloud account.

Implementation

For more information about RAM and how to create a RAM user, see [Introduction](#). To grant OSS access permissions to users by creating RAM policies, see [RAM Policy](#).

13.3 Access OSS with a temporary access token provided by STS

You can temporarily access OSS by using Security Token Service (STS) provided by Alibaba Cloud. Alibaba Cloud STS is a Web service that provides users with temporary access tokens. Using STS, you can grant an access credential with customized permissions and valid periods to third-party applications and federated users whose IDs are managed by you.

Scenarios

Users managed by your local identity system are referred to as federated users, for example, the users of your applications, local accounts owned by your enterprises, and third-party applications. Federated users may need to access your OSS resources directly. In addition, federated users can also include the users that are created by you and have access to your applications and resources in Alibaba Cloud.

For these users, you can use STS to manage the temporary access tokens for their Alibaba Cloud accounts (or RAM users). You can create temporary access credentials for federated users to grant OSS access permissions to them without providing your long term keys (such as logon password and AccessKeys) of your Alibaba Cloud accounts or RAM users to the federated users. The permissions and valid period of the credential can be customized. You do not need to revoke the permissions of the credential because it automatically becomes invalid after it expires.

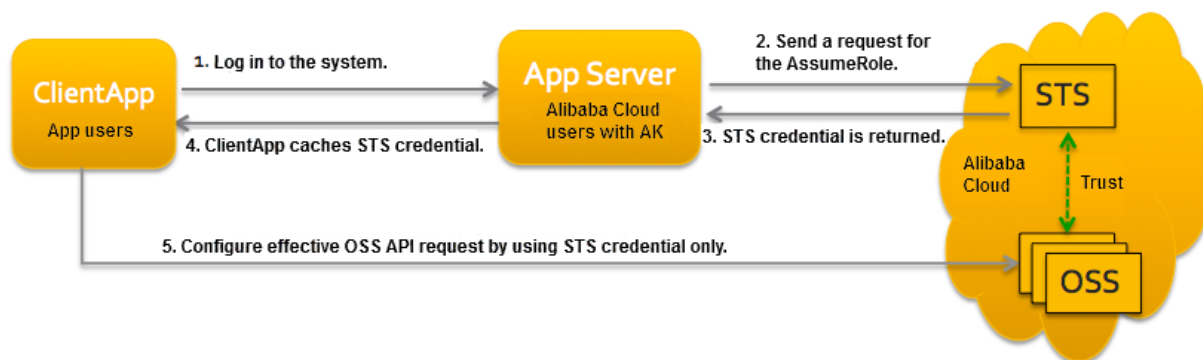
Credentials generated by STS include security tokens (SecurityToken) and temporary access keys (AccessKeyId and AccessKeySecret). You can use a temporary access key in the same way as you use the AccessKey of an Alibaba Cloud account or a RAM user to send a request. Each request sent to OSS must carry a security token.

Implementation

A mobile application is used as an example. Assume that you are a mobile application developer and try to use Alibaba Cloud OSS to store end user data for your app.

You must keep the data of each application user isolated to prevent the data of an application user from being obtained by other application users. You can use STS to authorize users so that they can directly access your OSS resources.

The following figure describes the process of using STS to grant OSS access to users.



1. An application user logs on to the application server. An application user is an end user of the application and has no relationship to an Alibaba Cloud account. The application server can be logged on by an application user. The application server must define the minimum access permission for each valid application user.
2. The application server request a security token from STS. Before calling STS, the application server must determine the minimum access permission for each application user (described in policy syntax) and the expiration time of the authorization. Then, the application server uses AssumeRole to obtain a security token which indicates a role.
3. STS returns a valid access credential to the application server. The credential includes a security token, a temporary access key (AccessKeyId and AccessKeySecret), and the expiration time.
4. The application server returns the access credential to the application user (ClientApp). The credential can be cached by the ClientApp. When the credential becomes invalid, the ClientApp must request a new valid access credential from the application server. For example, if the valid period of the returned access credential is an hour, the ClientApp can request the application server to update the access token every 30 minutes.

5. The ClientApp use the access credential in the local cache to request Alibaba Cloud service APIs. ECS perceives the STS access credential and uses STS to verify the credential so that it can correctly respond user's requests.

For more information about STS security tokens, role management, and role usage, see [Understand RAM roles](#). You can call the [AssumeRole](#) interface to obtain a valid access credential.

Procedure

Assume that a bucket named ram-test is used to store user data and it is required that STS should be used to grant permissions to a RAM user so that the user can access OSS buckets.

You can use OSS SDK and STS SDK together to access an OSS instance with a temporary access token provided by STS.

1. Create a RAM user.

- a. Log on to the [RAM console](#).
- b. In the RAM page, click Users.
- c. In the Users page, click Create User.
- d. In the Create User page, enter Logon Name and Display Name in the User Account Information area, select Programmatic Access for Access Mode, and then click OK.

RAM / Users / Create User

← Create User

* User Account Information

Logon Name [?] Display Name [?]

RAMtest @ [redacted].onaliyun.com Ramtest

+ Add User

Access Mode [?]

☐ Console Password Logon Users access the Alibaba Cloud console using the account and password.

☒ Programmatic Access Enable AccessKeyId and AccessKeySecret to support access through the API or other development tools.

OK Back

e. Select Permissions > Add Permissions.

RAM / Users / RAMTest@[redacted].onaliyun.com

← RAMTest@[redacted].onaliyun.com

Basic Information [?] Modify Basic Information

Username: RAMTest@[redacted].onaliyun.com [?] Copy

Display Name: RamTest

Note

Email Address

UID: [redacted]

Created: Dec 28, 2018, 14:29:30

Mobile Phone Number

Authentication Groups Permissions

Individual Group Permissions

Add Permissions

Applicable Scope of Permission	Policy	Policy Type	Note	Actions
--------------------------------	--------	-------------	------	---------

- f. In the Add Permissions page, add the AliyunSTSAssumeRoleAccess permission for the created RAM user.

Add Permissions

Principal
RAMTest@...onaliyun.com X

Select Policy

System Policy ▼ AliyunSTS 🔍 Selected (1) Clear

Policy Name	Note
AliyunSTSAssumeRoleAccess	Provides access to the API AssumeRole of Security Token Service(STS).

AliyunSTSAssumeRoleAccess X

Ok Cancel

**Note:**

Do not grant other permissions to the RAM user because it automatically obtains all permissions of a role when it acts as the role.

2. Create a permission policy.

- a. Log on to the [RAM console](#).
- b. In the RAM page, click Policies.
- c. Click Create Policy.
- d. In the Create Custom Policy page, enter the Policy Name and Note, and select Visualized or Script for Configure Mode.

For example, if you select Script and want to grant read only permissions, such as ListObjects and GetObject, to a RAM user named ram-test, add the following script in the Policy Document.

```
{
  "Version": " 1 ",
  "Statement": [
    {
      "Effect": " Allow ",
      "Action": [
        " oss : ListObject s ",
        " oss : GetObject "
      ],
      "Resource": [
        " acs : oss :*: *: ram - test ",
        " acs : oss :*: *: ram - test /*"
      ]
    }
  ]
}
```

```
}  
}  
}
```

← Create Custom Policy

Policy Name
Ramtest

Note
Eotest

Configuration Mode
☐ Visualized
☒ Script

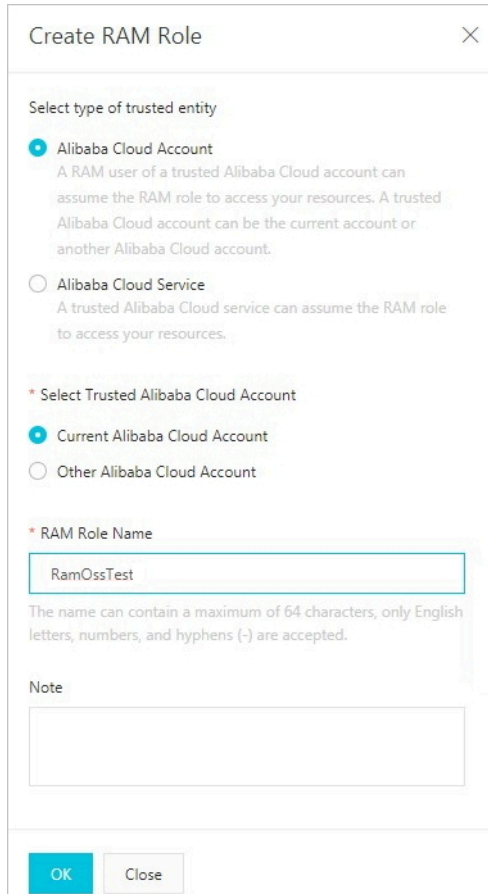
Policy Document
Import an existing system policy

```
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
    "Action": [  
      "oss:ListObjects",  
      "oss:GetObject"  
    ],  
    "Resource": [  
      "acs:oss:*:*:ram-test",  
      "acs:oss:*:*:ram-test/*"  
    ]  
  }  
}
```

OK Back

3. Create a role.

- a. Log on to the [RAM console](#).
- b. In the RAM page, click RAM Roles.
- c. In the RAM Roles page, click Create RAM Role.
- d. In the Create RAM Role page, enter the RAM Role Name (RamOssTest in this example), select the type of trusted entities and keep the default selection for Select Trusted Alibaba Cloud Account.



Create RAM Role

Select type of trusted entity

☒ Alibaba Cloud Account
A RAM user of a trusted Alibaba Cloud account can assume the RAM role to access your resources. A trusted Alibaba Cloud account can be the current account or another Alibaba Cloud account.

☐ Alibaba Cloud Service
A trusted Alibaba Cloud service can assume the RAM role to access your resources.

* Select Trusted Alibaba Cloud Account

☒ Current Alibaba Cloud Account

☐ Other Alibaba Cloud Account

* RAM Role Name

RamOssTest

The name can contain a maximum of 64 characters, only English letters, numbers, and hyphens (-) are accepted.

Note

OK Close

- e. Click Add Permissions on the right of the created role RamOssTest.
- f. In the Add Permissions page, select Custom Policy and add the policy Ramtest that you created in step 2.

After the policy is added, the page is shown as follows.

RAM / RAM Roles / RamOssTest

← RamOssTest

Basic Information

Role Name	RamOssTest	Created	Dec 28, 2018, 16:33:09
Note		ARN	acs:ram:::ssstest

Permissions Trust Policy Management

Add Permissions

Policy	Policy Type	Note	Actions
RAMtest	Custom Policy	For test	Remove Permission



Note:

ARN indicates the ID of the role that the RAM user acts.

4. Obtain an STS AK and security token through STS APIs

You can request STS through STS SDKs to obtain a security token. For more information about the installation and usage of STS SDKs, see [Installation](#).

The following code is described as an example to obtain a security token through STS Java SDK.

```
public class StsService Sample {
    public static void main (String [] args ) {
        String endpoint = " sts . aliyuncs . com ";
        String accessKeyId = "< access - key - id >";
        String accessKeySecret = "< access - key - secret >";
        String roleArn = "< role - arn >";
        String roleSessionName = " session - name ";
        String policy = "{\n" +
            "    \" Version \": \" 1 \",\n" +
            "    \" Statement \": [\n" +
            "        {\n" +
            "            \" Action \": [\n" +
            "                \" oss :*\n" +
            "            ],\n" +
            "            \" Resource \": [\n" +
            "                \" acs : oss :*:~:~:~*\n" +
            "            ],\n" +
            "            \" Effect \": \" Allow \"\n" +
            "        }\n" +
            "    ]\n" +
            "}";

        try {
            // Adds an endpoint . ( The STS endpoint
            // is directly used . The first two parameters are
            // left blank , indicating that the region ID is not
            // required .)
            DefaultProfile.addEndpoint ("", "", " Sts ",
            endpoint );
            // Constructs a default profile . ( The
            // parameter is left blank , indicating that the region
            // ID is not required .)
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```

        IClientProfile profile = DefaultProfile .
getProfile ("", accessKeyId , accessKeySecret );
        // Uses the constructed profile to
construct a client .
        DefaultAcsClient client = new DefaultAcs
Client ( profile );
        final AssumeRoleRequest request = new
AssumeRoleRequest ();
        request . setMethod ( MethodType . POST );
        request . setRoleArn ( roleArn );
        request . setRoleSessionName ( roleSessionName );
        Request . setPolicy ( policy ); // If the policy
is empty , the user obtains all permissions of
the role .
        Request . setDurationSeconds ( 1000l ); // Sets
the valid period of a credential .
        final AssumeRoleResponse response = client .
getAcsResponse ( request );
        System . out . println ( " Expiration : " + response .
getCredentials () . getExpiration () );
        System . out . println ( " Access Key Id : " +
response . getCredentials () . getAccessKeyId () );
        System . out . println ( " Access Key Secret : " +
response . getCredentials () . getAccessKeySecret () );
        System . out . println ( " Security Token : " +
response . getCredentials () . getSecurityToken () );
        System . out . println ( " RequestId : " + response .
getRequestId () );
    } catch ( ClientException e ) {
        System . out . println ( " Failed : " );
        System . out . println ( " Error code : " + e .
getErrorCode () );
        System . out . println ( " Error message : " + e .
getErrMsg () );
        System . out . println ( " RequestId : " + e .
getRequestId () );
    }
}
}

```

The parameters are described as follows:

- **AccessKeyId and AccessKey Secret:** Indicates the AK information about the RAM user.
- **RoleArn:** Indicates the ID of the role that the user acts.
- **RoleSessionName:** Indicates the name used to identify a temporary credential . We recommend you use different application user names to identify different credentials.
- **Policy:** Indicates the permission limits added to a user when the user acts as a role.



Note:

Policies are used to control the permissions of a temporary credential after the user acts as a role. The permission of a temporary credential is the intersection

of the role permissions and the policies. Policies are passed in to adjust the permissions more flexibly. For example, you can use policies to set different limits on the path where a file is upload for different users.

- **DurationSeconds:** Indicates the valid period (in seconds) of a temporary credential. The value of the parameter ranges from 900 to 3,600.

5. Access OSS using the STS AK and security token.

After obtaining the STS AK and security token, you can use the STS credential to construct a signed request.

```
// This example uses the endpoint China East 1 (
// Hangzhou ). Specify the actual endpoint based on your
// requirements .
String endpoint = " http :// oss - cn - hangzhou . aliyuncs .
com ";
// It is highly risky to log on with the
// AccessKey of an Alibaba Cloud account because the
// account has permissions on all the APIs in OSS
// . We recommend that you log on as a RAM user
// to access APIs or perform routine operations and
// maintenance . To create a RAM account , log on to
// https :// ram . console . aliyun . com .
String accessKeyId = "< yourAccess KeyId >";
String accessKeySecret = "< yourAccess KeySecret >";
String securityToken = "< yourSecurityToken >";

// After a user obtains a temporary STS credential ,
// the OSSClient is generated with the security token
// and temporary access key ( AccessKeyId and AccessKeySecret ) in the credential .
// Creates an OSSClient instance . Note that the STS
// credential generated in the preceding step is used .
OSSClient ossClient = new OSSClient ( endpoint , accessKeyId , accessKeySecret , securityToken );

// Performs OSS operations .

// Closes your OSSClient instance
ossClient . shutdown ();
```

14 Access and control

14.1 Overview

OSS provides multiple access control methods, including ACLs, RAM policies, and bucket policies, for users who access objects stored in buckets.

- **ACL:** OSS provides Access Control List (ACL) for access control. An ACL is set based on resources. You can set ACLs for buckets or objects. You can set an ACL for a bucket when you create the bucket or for an object when you upload the object to OSS. You can also modify the ACL for a created bucket or an uploaded object at anytime.
- **RAM Policy:** Resource Access Management (RAM) is a service provided by Alibaba Cloud for resource access control. RAM policies are configured based on users. By configuring RAM policies, you can manage multiple users in a centralized manner and control the resources that can be accessed by the users. For example, you can control the permission of a user so that the user can only read a specified bucket. A RAM user belongs to the Alibaba Cloud account under which it was created, and does not own any actual resources. That is, all resources belong to the corresponding Alibaba Cloud account.
- **Bucket Policy:** Bucket policies are configured based on resources. Compared with RAM policies, bucket policies can be directly configured on the graphical console. By configuring bucket policies, you can authorize users to access your bucket even you do not have permissions for RAM operations. By configuring bucket policies, you can grant access permissions to RAM users under other Alibaba Cloud accounts, and to anonymous users who access your resources from specified IP addresses or IP ranges.

14.2 Access control based on ACLs

OSS provides access control lists (ACLs) for you to control access permissions. ACLs are access policies that grant bucket and object access permissions to users. You can

set the ACL when creating a bucket or uploading an object, and modify the ACL for a created bucket or an uploaded object at any time.

**Note:**

For more information about ACL-based OSS APIs, see the following topics:

- Set the ACL for a bucket: [PutBucketACL](#)
- Obtain the ACL of a bucket: [GetBucketACL](#)
- Set the ACL for an object: [PutObjectACL](#)
- Obtain the ACL of an object: [GetObjectACL](#)

Bucket ACL

- Overview

Bucket ACLs are used to control access to buckets. You can set any of the following three types of ACLs for a bucket: public-read-write, public-read, and private, which are described in the following table.

ACL	Description	Access control
public-read-write	The public-read-write permission.	Anyone (including anonymous users) can perform read, write, and delete operations on the objects in the bucket . The bucket owner needs to pay the fees incurred by these operations. We recommend that you set this ACL with caution.
public-read	The public-read permission.	Only the bucket owner or authorized users can perform write and delete operations on the objects in the bucket. Other users (including anonymous users) can perform only read operations on the objects in the bucket.
private	The private permission.	Only the bucket owner or authorized users can perform read, write, and delete operations on the objects in the bucket. Without authorization, other users have no access to the objects in the bucket.

- Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
ossbrowser	Graphical tool, which is easy to operate
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Node.js SDK	
Ruby SDK	

Object ACL

- Overview

Object ACLs are used to control access to objects. You can set any of the following four types of ACLs for an object: private, public-read, public-read-write, and default. You can include the `x-oss-object-acl` field in the request header and send a PUT request through the PutObjectACL API to set the ACL for an object. Only the owner of a bucket can perform the PutObjectACL operation on the objects in the bucket.

The following table describes the four types of ACLs for objects.

ACL	Description	Access control
public-read-write	The public-read-write permission.	All users can read data from and write data to the object.
public-read	The public-read permission.	The object owner can read data from and write data to the object. Other users can only read data from the object.

ACL	Description	Access control
private	The private permission.	The object owner can read data from and write data to the object. Without authorization, other users have no access to the object.
default	The default permission.	The object inherits the ACL from the bucket, that is, the ACL of an object is the same as the ACL of the bucket where the object is stored.

**Note:**

- If no ACL is set for an object, the object uses the default ACL, indicating that the object has the same ACL as the bucket where the object is stored.
- If an ACL is set for an object, the object ACL takes precedence over the ACL of the bucket where the object is stored. For example, if the ACL of an object is set to public-read, all authenticated and anonymous users can read data from the object regardless of the bucket ACL.

· Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
ossbrowser	Graphical tool, which is easy to operate
ossutil	Command-line tool, which delivers good performance
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	

Reference

For more information about how to allow only specified users to access your objects, see the following topics:

- [Tutorial: Authorize a RAM user under another Alibaba Cloud account by adding a bucket policy](#)
- [Authorize another Alibaba Cloud account to access OSS resources](#)

14.3 Access control based on RAM Policy

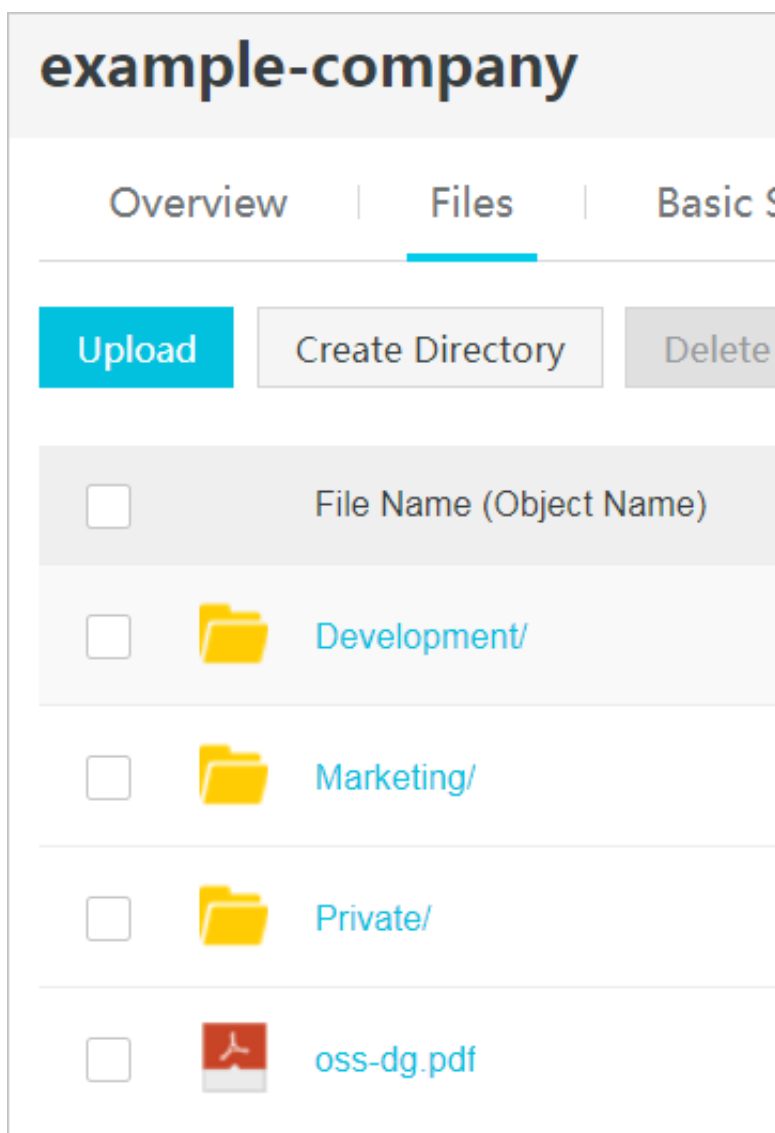
14.3.1 Tutorial: Use RAM Policy to control access to buckets and folders

This tutorial explains how to use RAM policies to grant and control user access to OSS buckets and folders.

In the example, we first create a storage space and folder, and then create access management (RAM) users using the Alibaba Cloud account, and grant these users incremental permissions to the created OSS storage space and folders by creating different RAM policies.

Buckets and folders

The data model structure of Alibaba Cloud OSS is flat instead of hierarchical. All objects (files) are directly related to their corresponding buckets. Therefore, OSS lacks the hierarchical structure of directories and subfolders as in a file system. However, you can emulate a folder hierarchy in the OSS console, where you can arrange and manage files by folders, as shown in the following figure.

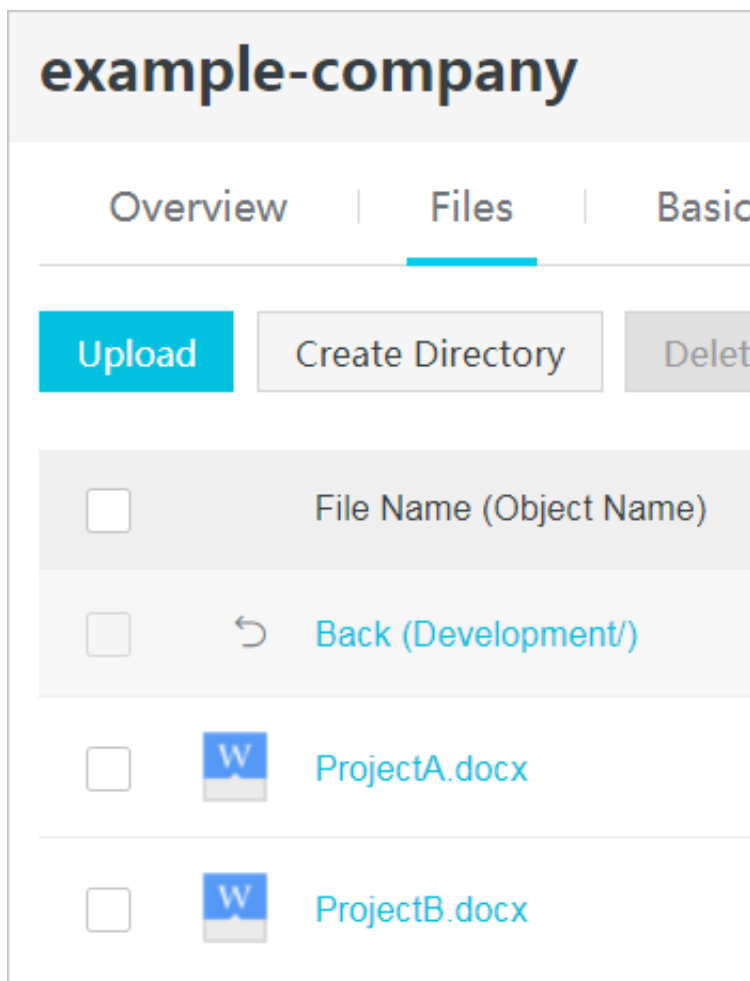


OSS is a distributed object storage service that uses a key-value pair format. Users retrieve the content of an object based on its unique key (object name). For example, the bucket named `example-company` has three folders: `Development`, `Marketing` and `Private`. This bucket also has one object `oss-dg.pdf`.

- When you create the `Development` folder, the console creates an object with the key `Development/`. Note that the key of a folder includes the delimiter `/`.
- When you upload an object named `ProjectA.docx` into the `Development` folder, the console uploads the object and sets its key to `Development/ProjectA.docx`.

In the key, `Development` is the prefix and `/` is the delimiter. You can retrieve a list of all objects with a specific prefix and delimiter from a bucket. In the console,

you click the `Development` folder, and the console lists the objects in the folder, as shown in the following figure.

**Note:**

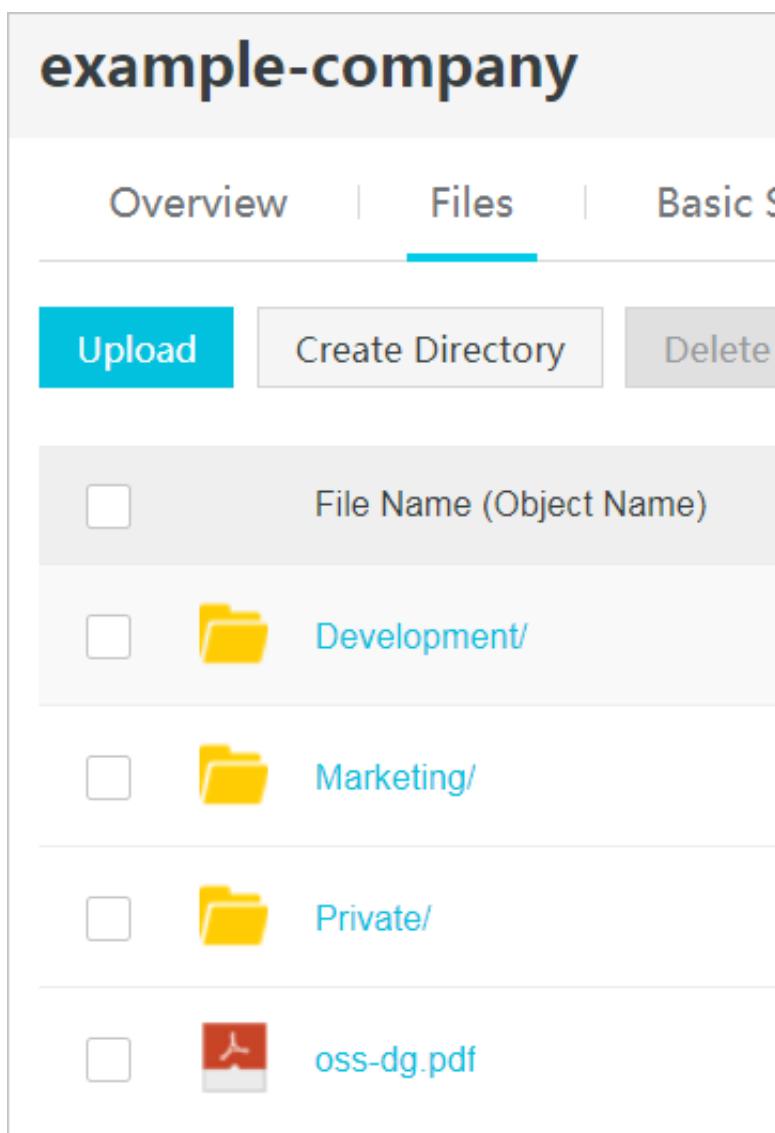
When the console lists the `Development` folder in the `example-company` bucket, it sends OSS a request which specifies the prefix `Development` and the delimiter `/`. The console responds with a folder list the same as that in the file system. The preceding example shows that the bucket `example-company` has two objects with the key `Development / Alibaba Cloud . pdf`, `Development / ProjectA . docx`, and `Development / ProjectB . docx`.

The console uses object keys to resemble a logical hierarchy. When you create a logical hierarchy of objects, you can manage access to individual folders, as described later in this tutorial.

Before going into the tutorial, you also need to know the concept: root-level bucket content. Suppose the `example-company` bucket has the following objects:

- Development/Alibaba Cloud.pdf
- Development/ProjectA.docx
- Development/ProjectB.docx
- Marketing/data2016.xlsx
- Marketing/data2016.xlsx
- Private/2017/images.zip
- Private/2017/promote.pptx
- oss-dg.pdf

These object keys resemble a logical hierarchy with *Development*, *Marketing*, and *Private* as root-level folders and *oss-dg.pdf* as a root-level object. When you click the bucket name in the OSS console, the console shows the first-level prefixes and a delimiter (*Development /*, *Marketing /*, and *Private /*) as root-level folders. The object key *oss - dg . pdf* does not have a prefix, so it appears as a root-level item.



Request and response of OSS

Before granting permissions, we need to understand what request the console sends to OSS when a user clicks a bucket name, the response OSS returns, and how the console interprets the response.

When a user clicks a bucket name, the console sends the [GetBucket](#) request to OSS.

This request includes the following parameters:

- `prefix` with an empty string as its value.
- `delimiter` with `/` as its value.

An example request is as follows:

```
GET /? prefix =& delimiter =/ HTTP / 1 . 1
Host : example - company . oss - cn - hangzhou . aliyuncs . com
Date : Fri , 24 Feb 2012 08 : 43 : 27 GMT
```

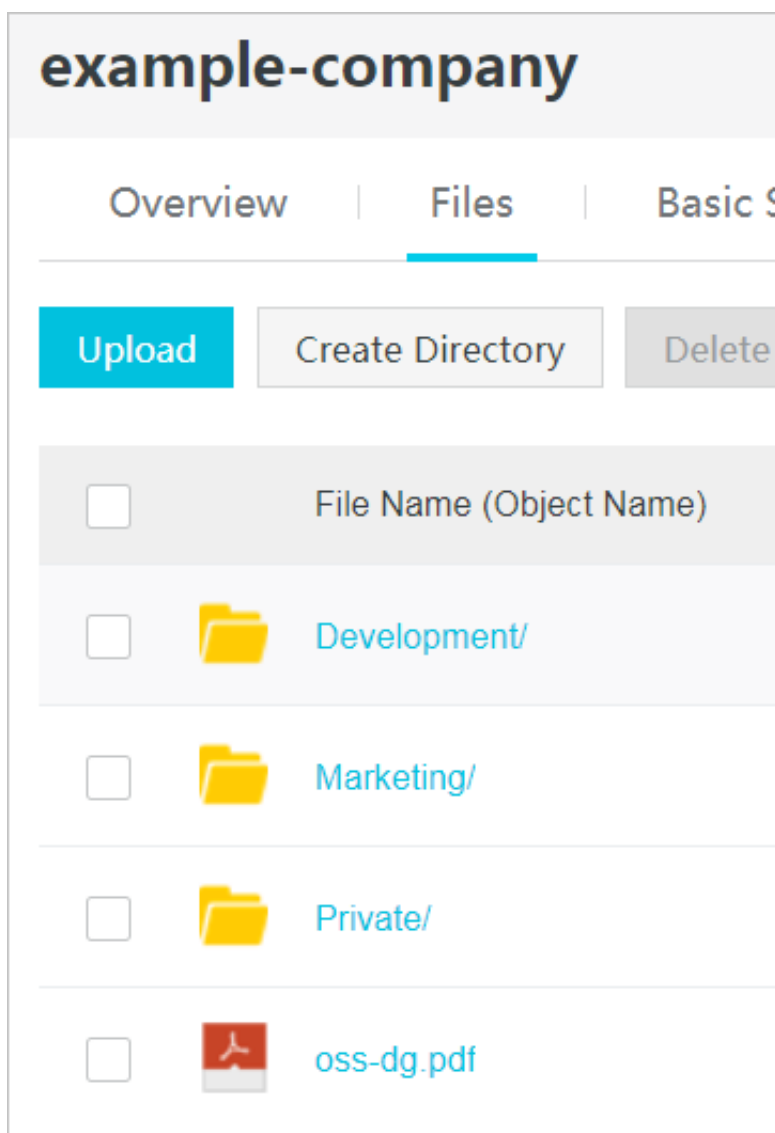
```
Authorization : OSS qn6qrrqxo2 oawuk53otf jbyc : DNrn7xHk3
sgysx7I8U9 I9IY1vY =
```

OSS returns a response that includes the `ListBucket Result` element:

```
HTTP / 1 . 1 200 OK
x - oss - request - id : 534B371674 E88A4D8906 008B
Date : Fri , 24 Feb 2012 08 : 43 : 27 GMT
Content - Type : applicatio n / xml
Content - Length : 712
Connection : keep - alive
Server : AliyunOSS
<? xml version =" 1 . 0 " encoding =" UTF - 8 "? >
< ListBucket Result xmlns =± http :// doc . oss - cn - hangzhou .
aliyuncs . com ±>
< Name > example - company </ Name >
< Prefix ></ Prefix >
< Marker ></ Marker >
< MaxKeys > 100 </ MaxKeys >
< Delimiter ></ Delimiter >
  < IsTruncate d > false </ IsTruncate d >
  < Contents >
    < Key > oss - dg . pdf </ Key >
    ...
  </ Contents >
  < CommonPref ixes >
    < Prefix > Developmen t </ Prefix >
  </ CommonPref ixes >
    < CommonPref ixes >
      < Prefix > Marketing </ Prefix >
    </ CommonPref ixes >
    < CommonPref ixes >
      < Prefix > Private </ Prefix >
    </ CommonPref ixes >
  </ ListBucket Result >
```

The key `oss - dg . pdf` does not contain the `/` delimiter, so OSS returns the key in the `< Contents />` element. All other keys in the bucket `example-company` contain the `/` delimiter, so OSS groups these keys and returns a `CommonPref ixes /` element for each of the prefix values `Developmen t /`, `Marketing /`, and `Private /`. The `CommonPrefixes/` element includes a substring of the key name, which starts from the beginning of the key name and ends with (but does not include) the first occurrence of the specified `/` delimiter.

The console interprets this result and displays the root-level items as follows:



Now, if a user clicks the *Development* folder, the console sends the [GetBucket](#) request to OSS. This request includes the following parameters:

- `prefix` with `Development /` as its value
- `delimiter` with `/` as its value.

An example request is as follows:

```
GET /? prefix = Development /& delimiter =/ HTTP / 1 . 1
Host : oss - example . oss - cn - hangzhou . aliyuncs . com
Date : Fri , 24 Feb 2012 08 : 43 : 27 GMT
Authorization : OSS qn6qrrqxo2 oawuk53otf jbyc : DNrnrx7xHk3
sgysx7I8U9 I9IY1vY =
```

In response, OSS returns the object keys that begin with the specified prefix:

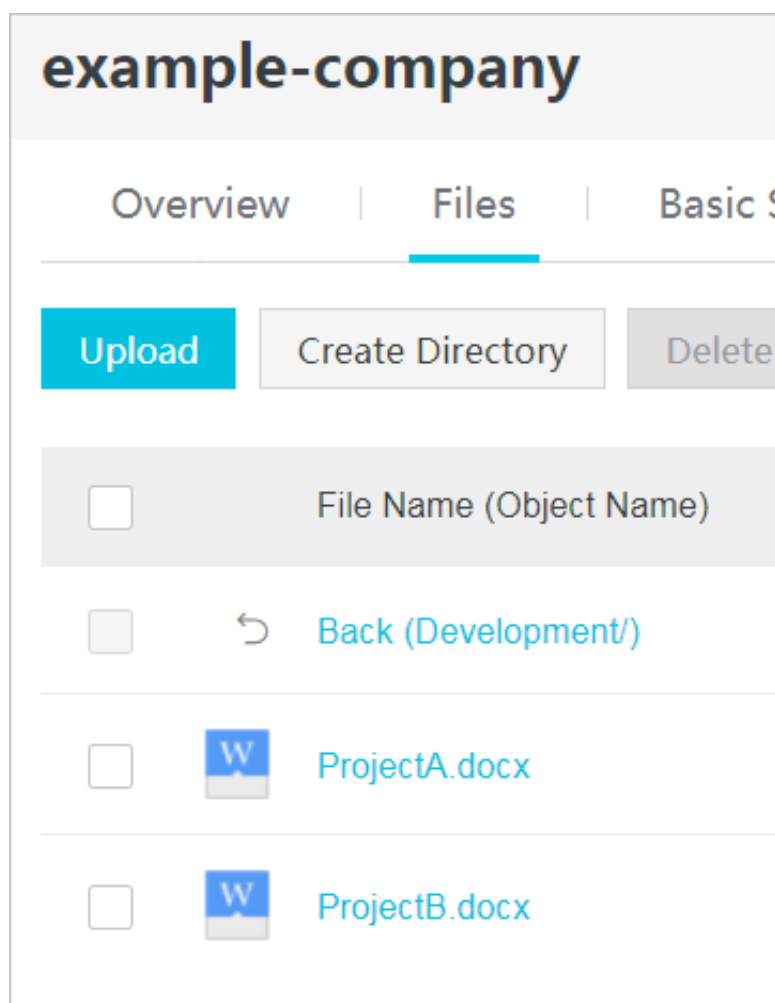
```
HTTP / 1 . 1 200 OK
x - oss - request - id : 534B371674 E88A4D8906 008B
Date : Fri , 24 Feb 2012 08 : 43 : 27 GMT
Content - Type : applicatio n / xml
```

```

Content - Length : 712
Connection : keep - alive
Server : AliyunOSS
<? xml version="1.0" encoding="UTF-8" ? >
< ListBucket Result xmlns="http://doc.oss-cn-hangzhou.aliyuncs.com" >
  < Name > example-company </ Name >
  < Prefix > Development /</ Prefix >
  < Marker ></ Marker >
  < MaxKeys > 100 </ MaxKeys >
  < Delimiter ></ Delimiter >
    < IsTruncated > false </ IsTruncated >
    < Contents >
      < Key > ProjectA.docx </ Key >
      ...
    </ Contents >
    < Contents >
      < Key > ProjectB.docx </ Key >
      ...
    </ Contents >
  </ ListBucket Result >

```

The console interprets this result and displays the object keys as follows:



Tutorial example

The tutorial example is as follows:

- You create a bucket `example-company` and then add three folders (`Development` , `Marketing` , and `Private`) into it.
- You have two users, Anne and Leo. You want Anne to access only the `Development` folder and Leo to access only the `Marketing` folder, and you want to keep the `Private` folder private. In the tutorial example, you manage access by creating Alibaba Cloud identity and Resource Access Management (RAM) users (Anne and Leo) and granting them the necessary permissions.
- RAM also supports the creation of user groups and granting of group-level permissions that apply to all users in the group. This helps you better manage and control permissions. For this example, both Anne and Leo need some common permissions. You also create a group named `Staff` and then add both Anne and Leo to the `Staff` group. You first grant permissions by attaching a group policy to the `Staff` group. Then you add user-specific permissions by attaching policies to specific users.

**Note:**

The tutorial uses `example-company` as the bucket name, Anne and Leo as the RAM users, and `Staff` as the group name. Because Alibaba Cloud OSS requires that bucket names be globally unique, you must replace the bucket name with your own unique bucket name.

Prepare for the tutorial

In this example, you use your Alibaba Cloud account to create RAM users. Initially , these users have no permissions. You incrementally grant these users permissions to perform specific OSS operations. To test these permissions, you log on to the console with each user' s credentials. As you incrementally grant permissions as an Alibaba Cloud account owner and test permissions as a RAM user, you have to log on and log off, each time using different credentials. You can perform this testing with one browser, but the process is more efficient if you can use two different browsers : one browser to connect to the Alibaba Cloud console with your primary account credentials and the other to connect with the RAM user credentials.

To log on to the Alibaba Cloud console with your Alibaba Cloud account credentials , go to <https://account.alibabacloud.com/login/login.htm>. RAM users cannot log on by using the same link. They must use the RAM user logon link. As the owner of the Alibaba Cloud account, you can provide this logon link to your users.

**Note:**

For more information about RAM, see [Log on with a RAM user account](#).

Provide a logon link for RAM users

1. Log on to the [RAM console](#) with your Alibaba Cloud account credentials.
2. In the left-side navigation pane, click Dashboard.
3. Find the URL after RAM User Logon Link: You will provide this URL to RAM users to log on to the console with their RAM user name and password.

Step 1. Create a bucket

In this step, you log on to the OSS console with your primary account credentials, create a bucket, add folders (*Development* , *Marketing* , *Private*) to the bucket, and upload one or two sample documents in each folder.

1. Log on to the [OSS console](#).
2. Create a bucket named example-company.

For detailed procedures, see [Create a bucket](#) in the OSS Console User Guide.

3. Upload a file to the bucket.

This example assumes that you upload the file `oss - dg . pdf` at the root level of the bucket. You can upload your own file with a different file name.

For detailed procedures, see [Upload files](#) in the OSS Console User Guide.

4. Create three folders named *Development* , *Marketing* , and *Private* .

For detailed procedures, see [Create a folder](#) in the OSS Console User Guide.

5. Upload one or two files to each folder.

This example assumes that you upload objects in the bucket with the following object keys:

- Development/Alibaba Cloud.pdf
- Development/ProjectA.docx
- Development/ProjectB.docx
- Marketing/data2016.xlsx
- Marketing/data2016.xlsx
- Private/2017/images.zip
- Private/2017/promote.pptx
- oss-dg.pdf

Step 2. Create RAM users and a group

In this step, you use the RAM console to add two RAM users, Anne and Leo, to your Alibaba Cloud account. You also create a group named Staff, and then add both users to the group.



Note:

In this step, do not attach any policies that grant permissions to these users. In the following steps, you will incrementally grant permissions.

For detailed procedures on creating a RAM user, see [Create a RAM user](#) in the RAM Quick Start. Remember to create a logon password for each RAM user.

For detailed procedures on creating a group, see the Create a group section of [Groups](#) in the RAM User Guide.

Step 3: Verify that RAM users have no permissions

If you use two browsers, now you can use the other one to log on to the console by using one of the RAM user credentials.

1. Open the RAM user logon page, and log on to the RAM console with Anne's or Leo's credentials.
2. Open the OSS console.

You find no buckets in the console, which means that Anne does not have any permissions on the bucket example-company.

Step 4: Grant group-level permissions

We want both Anne and Leo to have the access and ability to perform the following tasks:

- List all buckets owned by the Alibaba Cloud account.

To do this, Anne and Leo must have permission for the `oss : ListBucket s` action.

- List root-level items, folders, and objects, in the example-company bucket.

To do this, Anne and Leo must have permission for the `oss : ListObject s` action on the example-company bucket.

Step 4.1: Grant permissions to list all buckets

In this step, you create a policy that grants users minimum permissions. With the minimum permissions, users can list all buckets owned by the Alibaba Cloud account . You also attach the policy to the Staff group, so you grant the group permission to get a list of buckets owned by the primary account.

1. Log on to the [RAM console](#) with your Alibaba Cloud account credentials.
2. Create a policy `AllowGroupToSeeBucketListInConsole`.
 - a. From the left-side navigation pane, click Policies, and then click Create Authorization Policy.
 - b. Click Blank Template.
 - c. In the Authorization Policy Name field, enter `AllowGroupToSeeBucketListInConsole`.
 - d. In the Policy Content field, copy and paste the following policy.

```
{
  "Version": " 1 ",
  "Statement": [
    {
      "Effect": " Allow ",
      "Action": [
        " oss : ListBucket s "
      ],
      "Resource": [
        " acs : oss :*:~:*"
      ]
    }
  ]
}
```

```
}
```

**Note:**

- A policy is in JSON format. The fields in the policy are described as follows:
 - **Statement:** This attribute is an array of objects, and each object describes a permission using a collection of name value pairs.
 - **Effect:** This attribute value determines whether a specific permission is allowed or denied.
 - **Action:** This attribute specifies the type of access. In the policy, the `oss:ListBucket` is a predefined OSS action, which returns a list of all buckets owned by the authenticated sender.
- We recommend you use [RAM policy editor](#) to generate a RAM policy quickly. For more information about RAM policies, see [How to create a RAM policy](#).

3. Attach the `AllowGroupToSeeBucketListInConsole` policy to the Staff group.

For detailed procedures on attaching a policy, see the [Attach policies to a RAM group](#) section of Attach policies to a RAM user in the RAM Quick Start.

You can attach policies to RAM users and groups in the RAM console. In this example, we attach the policy to the group, because we want both Anne and Leo to be able to list the buckets.

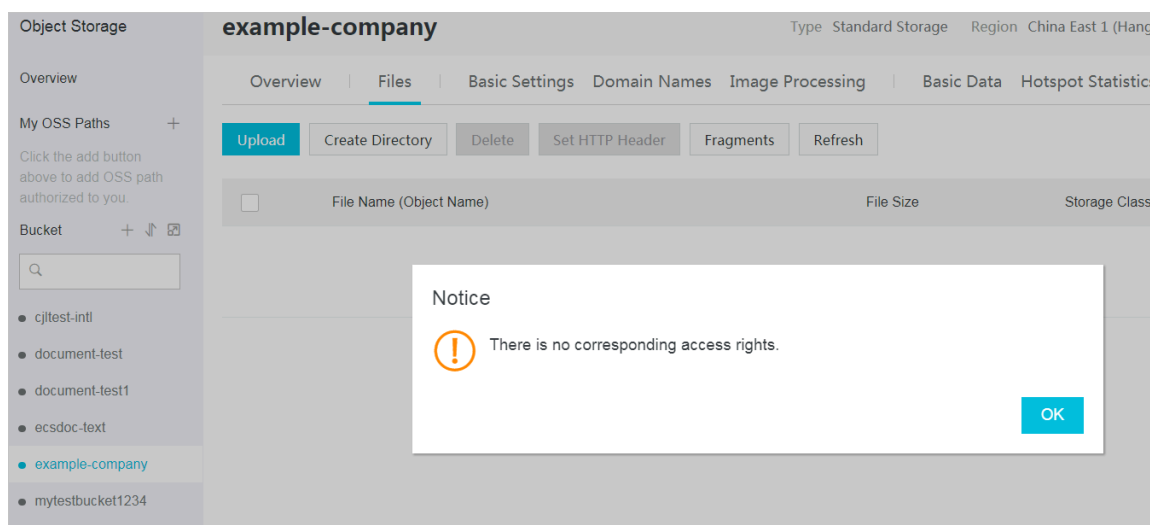
4. Test the permission.

- a. Open the RAM user logon page, and log on to the RAM console with Anne' s or Leo' s credentials.
- b. Open the OSS console.

The console lists all of the buckets.

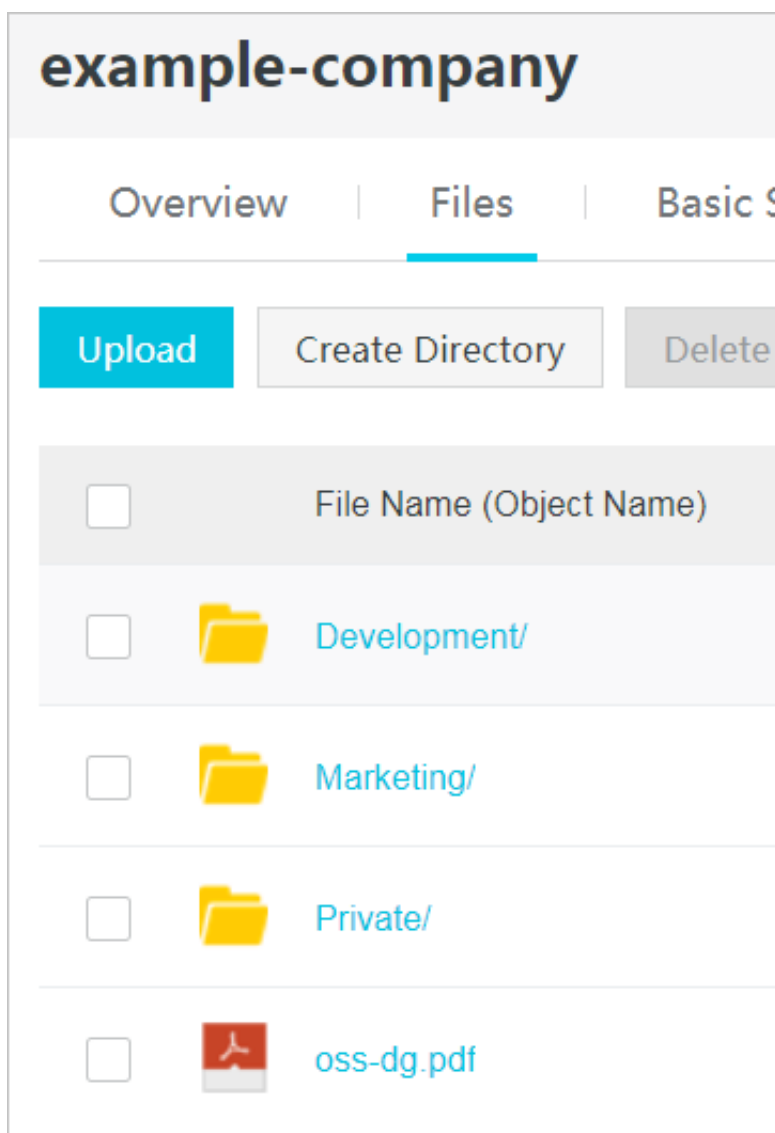
- c. Click the example-company bucket, and then click the Files tab.

A message box is displayed, indicating that you have no corresponding access rights.



Step 4.2: Grant permissions to list root-level content of a bucket

In this step, you grant permissions to allow all users to list all the items in the bucket example-company. When users click the example-company in the OSS console, they can see the root-level items in the bucket.



1. Log on to the [RAM console](#) with your Alibaba Cloud account credentials.
2. Replace the existing policy `AllowGroup ToSeeBuckets ListInConsole` that is attached to the Staff group with the following policy. The following policy also allows the `oss : ListObjects` operation. Remember to replace `example-company` in the policy Resource with the name of your bucket.

For detailed procedures, see the [Modify a custom authorization policy](#) section of Authorization policies in the RAM User Guide. Note that you can modify a RAM policy a maximum of five times. If this is exceeded, you must delete the policy, create a new one, and then attach the policy to the Staff group again.

```
{
  "Version": "1",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```

        "oss : ListBucket s ",
        "oss : GetBucketA cl "
    ],
    "Resource ": [
        "acs : oss :*:~:*"
    ],
    "Condition ": {}
},
{
    "Effect ": " Allow ",
    "Action ": [
        "oss : ListObject s "
    ],
    "Resource ": [
        "acs : oss :*:~:* example - company "
    ],
    "Condition ": {
        "StringLike ": {
            "oss : Prefix ": [
                ""
            ],
            "oss : Delimiter ": [
                "/"
            ]
        }
    }
}
]
}

```



Note:

- To list bucket content, users need permission to call the `oss : ListObject s` operation. To make sure that they see only the root-level content, we add a condition that users must specify an empty prefix in the request, that is, they cannot click any of the root-level folders. We also add a condition to require folder-style access by requiring user requests to include the delimiter parameter with the value `/`.
- When a user logs on to the OSS console, the console checks the user's identities for access to OSS. To support bucket operations in the console, we also need to add the `oss : GetBucketA cl` operation.

3. Test the updated permissions.

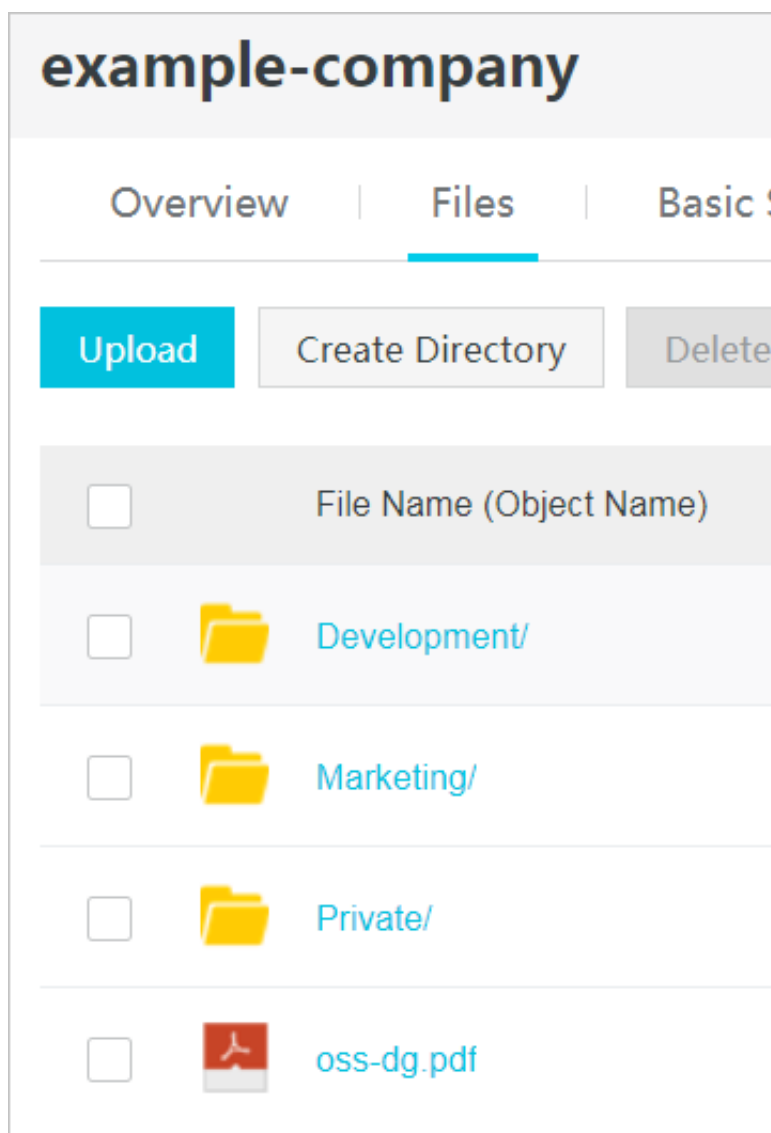
- a. Open the RAM user logon page, and log on to the RAM console with Anne' s or Leo' s credentials.

- b. Open the OSS console.

The console lists all of the buckets.

- c. Click the example-company bucket, and then click the Files tab.

The console lists all the root-level items.



- d. Click any of the folders or the `object oss - dg . pdf`.

A message box is displayed, indicating that you have no corresponding access rights.

Summary of the group policy

The result of the group policy that you have added is to grant the RAM users Anne and Leo the following minimum permissions:

- The ability to list all buckets owned by the primary account.
- The ability to see all root-level items in the example-company bucket.

However, they still have limited access. In the following section, we grant further user-specific permissions including:

- Provide Anne the ability to get and put objects in the `Development` folder.
- Provide Bob the ability to get and put objects in the `Finance` folder.

For user-specific permissions, you attach a policy to the specific user, not to the entire group. In the following section, you grant Anne permission to work within the `Development` folder. You can repeat the steps to grant similar permission to Leo to work in the `Finance` folder.

Step 5: Grant RAM user Anne specific permissions

In this step, we grant additional permissions to Anne so that she can see the content of the `Development` folder and get and put objects in the folder.

Step 5.1: Grant RAM user Anne permission to list the `Development` folder content

For Anne to list the `Development` folder content, you must attach a policy to her that grants permission for the `oss : ListObjects` action on the example-company bucket, and includes the condition that user must specify the prefix `Development /` in the request.

1. Log on to the [RAM console](#) with your Alibaba Cloud account credentials.
2. Create a policy `AllowListBucketsIfSpecificPrefixIsIncluded` that grants the RAM user Anne permission to list the `Development` folder content.
 - a. From the left-side navigation pane, click Policies, and then click Create Authorization Policy.
 - b. Click Blank Template.
 - c. In the `Authorization Policy Name` field, enter `AllowListBucketsIfSpecificPrefixIsIncluded`.
 - d. In the `Policy Content` field, copy and paste the following policy.

```
{  
  "Version": "1",
```

```

" Statement ": [
{
  " Effect ": " Allow ",
  " Action ": [
    " oss : ListObject s "
  ],
  " Resource ": [
    " acs : oss :*: *: example - company "
  ],
  " Condition ": {
    " StringLike ": {
      " oss : Prefix ": [
        " Developmen t /*"
      ]
    }
  }
}
]
}

```

3. Attach the policy to the RAM user Anne.

For detailed procedures on attaching a policy, see [Attach policies to a RAM user](#) in the RAM Quick Start.

4. Test Anne' s permissions.

- a. Open the RAM user logon page, and log on to the RAM console with Anne' s credentials.
- b. Open the OSS console. The console lists all of the buckets.
- c. Click the example-company bucket, and then click the Files tab. The console lists all the root-level items.
- d. Click the `Developmen t /` folder. The console lists the objects in the folder.

Step 5.2 Grant RAM User Anne permissions to get and put objects in the `Developmen t` folder

For Anne to get and put objects in the `Developmen t` folder, you must grant her permission to call the `oss : GetObject` and `oss : PutObject` actions, and includes the condition that user must specify the prefix `Developmen t /` in the request.

1. Log on to the [RAM console](#) with your Alibaba Cloud account credentials.
2. Replace the policy `AllowListB ucketsIfSp ecificPref ixIsInclud ed` you created in the previous step with the following policy.

For detailed procedures, see the [Modify a custom authorization policy](#) section of Authorization policies in the RAM User Guide. Note that you can modify a RAM

policy a maximum of five times. If this is exceeded, you must delete the policy, created a new one, and then attach the policy to the user again.

```
{
  "Version": " 1 ",
  "Statement": [
    {
      "Effect": " Allow ",
      "Action": [
        " oss : ListObject s "
      ],
      "Resource": [
        " acs : oss :*:*: example - company "
      ],
      "Condition": {
        "StringLike": {
          " oss : Prefix ": [
            " Developmen t /*"
          ]
        }
      }
    },
    {
      "Effect": " Allow ",
      "Action": [
        " oss : GetObject ",
        " oss : PutObject ",
        " oss : GetObjectA cl "
      ],
      "Resource": [
        " acs : oss :*:*: example - company / Developmen t /*"
      ],
      "Condition": {}
    }
  ]
}
```



Note:

When a user logs on to the OSS console, the console checks the user' s identities for access to the OSS service. To support bucket operations in the console, we also need to add the `oss : GetObjectA cl` action.

3. Test the updated policy.

- a. Open the RAM user logon page, and log on to the RAM console with Anne' s credentials.
- b. Open the OSS console.

The console lists all of the buckets.

- c. In the OSS console, verify that Anne can now add an object and download an object in the `Developmen t` folder.

Step 5.3 Explicitly deny RAM user Anne permissions to any other folders in the bucket

RAM user Anne can now list the root-level content in the example-company bucket, and get and put objects in the `Development` folder. If you want to strictly restrict the access permissions, you can explicitly deny Anne's access to any other folders in the bucket. If other policies grant Anne's access to any other folders in the bucket, this explicit policy overrides those permissions.

You can add the following statement to the RAM user Anne's policy `AllowListBucketsIfSpecificPrefixIsIncluded`. The following statement requires all requests that Anne sends to OSS to include the prefix parameter, and the parameter value can be either `Development/*` or an empty string.

```
{
  "Effect": "Deny",
  "Action": [
    "oss:ListObjects"
  ],
  "Resource": [
    "acs:oss:*:*:example-company"
  ],
  "Condition": {
    "StringNotLike": {
      "oss:Prefix": [
        "Development/*",
        ""
      ]
    }
  ]
}
```

Follow the preceding step to update the policy `AllowListBucketsIfSpecificPrefixIsIncluded` that you created for RAM user Anne. Copy and paste the following policy to replace the existing one.

```
{
  "Version": "1",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "oss:ListObjects"
      ],
      "Resource": [
        "acs:oss:*:*:example-company"
      ],
      "Condition": {
        "StringLike": {
          "oss:Prefix": [
            "Development/*"
          ]
        }
      }
    },
    {
      "Effect": "Allow",
```

```

    " Action ": [
      " oss : GetObject ",
      " oss : PutObject ",
      " oss : GetObjectAcl "
    ],
    " Resource ": [
      " acs : oss :*:example-company/Development/* "
    ],
    " Condition ": {}
  },
  {
    " Effect ": " Deny ",
    " Action ": [
      " oss : ListObjects "
    ],
    " Resource ": [
      " acs : oss :*:example-company "
    ],
    " Condition ": {
      " StringNotLike ": {
        " oss : Prefix ": [
          " Development/*",
          ""
        ]
      }
    }
  }
]
}

```

Step 6: Grant RAM user Leo specific permissions

Now you want to grant Leo permission to the *Marketing* folder. Follow the steps you used earlier to grant permissions to Anne, but replace the *Development* folder with the *Marketing* folder. For detailed procedures, see Step 5: Grant RAM user Anne specific permissions.

Step 7: Secure the Private folder

In this example, you have only two users. In this example, you have only two users. You have granted all the minimum required permissions at the group level. In addition, you have granted user-level permissions only when you really need permissions at the individual user level. This approach helps minimize the effort of managing permissions. As the number of users increases, we want to make sure that we do not accidentally grant a user permission to the *Private* folder. Therefore we need to add a policy that explicitly denies access to the *Private* folder. An explicit denial overrides any other permissions. To make sure that the *Private* folder remains private, you can add the following two deny statements to the group policy:

- Add the following statement to explicitly deny any action on resources in the `Private` folder (`example - company / Private /*`).

```
{
  " Effect ": " Deny ",
  " Action ": [
    " oss :*"
  ],
  " Resource ": [
    " acs : oss :*: *: example - company / Private /*"
  ],
  " Condition ": {}
}
```

- You also deny permission for the `ListObject s` action when the request specifies the `Private / prefix` . In the console, if Anne or Leo clicks the `Private` folder, this policy causes OSS to return an error response.

```
{
  " Effect ": " Deny ",
  " Action ": [
    " oss : ListObject s "
  ],
  " Resource ": [
    " acs : oss :*: *: *"
  ],
  " Condition ": {
    " StringLike ": {
      " oss : Prefix ": [
        " Private /"
      ]
    }
  }
}
```

- Replace the Staff group policy `AllowGroup ToSeeBucke tListInCon sole` with an updated policy that includes the preceding deny statements. After the updated policy is applied, none of the users in the group can access the `Private` folder in your bucket.
1. Log on to the [RAM console](#) with your Alibaba Cloud account credentials.
 2. Replace the existing policy `AllowGroup ToSeeBucke tListInCon sole` that is attached to the Staff group with the following policy. Remember to replace `example-company` in the policy `Resource` with the name of your bucket.

```
{
  " Version ": " 1 ",
  " Statement ": [
    {
      " Effect ": " Allow ",
      " Action ": [
        " oss : ListBucket s ",
        " oss : GetBucketA cl "
      ]
    }
  ]
}
```

```

    ],
    "Resource ": [
        "acs : oss :*:~:*"
    ],
    "Condition ": {}
},
{
    "Effect ": " Allow ",
    "Action ": [
        "oss : ListObject s "
    ],
    "Resource ": [
        "acs : oss :*:~:* example - company "
    ],
    "Condition ": {
        "StringLike ": {
            "oss : Prefix ": [
                ""
            ],
            "oss : Delimiter ": [
                "/"
            ]
        }
    }
},
{
    "Effect ": " Deny ",
    "Action ": [
        "oss :*"
    ],
    "Resource ": [
        "acs : oss :*:~:* example - company / Private /*"
    ],
    "Condition ": {}
},
{
    "Effect ": " Deny ",
    "Action ": [
        "oss : ListObject s "
    ],
    "Resource ": [
        "acs : oss :*:~:*"
    ],
    "Condition ": {
        "StringLike ": {
            "oss : Prefix ": [
                " Private /"
            ]
        }
    }
}
]
}

```

Cleanup

After you finish the tutorial, remove the users Anne and Leo in the RAM console.

For detailed procedures, see [Delete a RAM user](#) section of Users in the RAM User Guide.

To avoid any unnecessary charges, delete the objects and the bucket that you created for this tutorial.

14.3.2 RAM policy

Resource Access Management (RAM) is a service provided by Alibaba Cloud for resource access control. RAM policies are configured based on users. By configuring RAM policies, you can manage multiple users in a centralized manner and control the resources that can be accessed by the users. For example, you can control the permission of a user so that the user can only read a specified bucket. A RAM user belongs to the Alibaba Cloud account under which it was created, and does not own any actual resources. That is, all resources belong to the corresponding Alibaba Cloud account.

Policy examples

- Policies that grant full permissions

A policy that grants full permissions allows applications to perform all operations on OSS.



Warning:

We recommend that you do not use a policy that grants full permissions for mobile applications because it is not secure.

```
{
  "Statement": [
    {
      "Action": [
        "oss:*"
      ],
      "Effect": "Allow",
      "Resource": ["acs:oss:*:*:*"]
    }
  ],
  "Version": "1"
}
```

Operations on OSS	Result
List all created buckets.	Success
Upload an object without a prefix, such as text.txt.	Success
Download an object without a prefix, such as text.txt.	Success

Operations on OSS	Result
Upload an object with a prefix, such as user1/test.txt.	Success
Download an object with a prefix, such as user1/test.txt.	Success
List objects without prefixes, such as test.txt.	Success
List objects with prefixes, such as user1/test.txt.	Success

- Read-only policies for all objects

A read-only policy indicates that an application can list and download all objects in the bucket `app - base - oss`.

```
{
  "Statement": [
    {
      "Action": [
        "oss:GetObject",
        "oss:ListObjects"
      ],
      "Effect": "Allow",
      "Resource": ["acs:oss:*:*:app-base-oss/*", "acs:oss:*:*:app-base-oss"]
    }
  ],
  "Version": "1"
}
```

Operations on OSS	Result
List all created buckets.	Failed
Upload an object without a prefix, such as test.txt.	Failed
Download an object without a prefix, such as test.txt.	Success
Upload an object with a prefix, such as user1/test.txt.	Failed
Download an object with a prefix, such as user1/test.txt.	Success
List objects without prefixes, such as test.txt.	Success
List objects with prefixes, such as user1/test.txt.	Success

- Read-only policies for objects with a specified prefix

This kind of policy indicates that an application can list and download objects with the `user1 /prefix` in the bucket `app - base - oss` but cannot download objects with other prefixes. Using this kind of policy, you can isolate applications with different prefixes in a bucket.

```
{
  "Statement": [
    {
      "Action": [
        "oss : GetObject ",
        "oss : ListObject s "
      ],
      "Effect": "Allow ",
      "Resource": [
        "acs : oss :*:*: app - base - oss / user1 /",
        "acs : oss :*:*: app - base - oss "
      ]
    }
  ],
  "Version": " 1 "
}
```

Operations on OSS	Result
List all created buckets.	Failed
Upload an object without a prefix, such as text.txt.	Failed
Download an object without a prefix, such as text.txt.	Failed
Upload an object with a prefix, such as user1/test.txt.	Failed
Download an object with a prefix, such as user1/test.txt.	Success
List objects without prefixes, such as test.txt.	Success
List objects with prefixes, such as user1 /test.txt.	Success

- Write-only policies for all objects

A write-only policy for all objects indicates that an application can upload objects to the bucket `app - base - oss`.

```
{
  "Statement": [
    {
      "Action": [
        "oss : PutObject "
      ]
    }
  ]
}
```



```

    ],
    "Effect ": " Allow ",
    "Resource ": ["acs : oss :*: *: app - base - oss /*", "acs
: oss :*: *: app - base - oss "]
  }
],
"Version ": " 1 "
}

```

Operations on OSS	Result
List all created buckets.	Failed
Upload an object without a prefix, such as text.txt.	Success
Download an object without a prefix, such as text.txt.	Failed
Upload an object with a prefix, such as user1/test.txt.	Success
Download an object with a prefix, such as user1/test.txt.	Success
List objects without prefixes, such as test.txt.	Success
List objects with prefixes, such as user1 /test.txt.	Success

- Write-only policies for objects with a specified prefix

This kind of policy indicates that an application can upload objects with the prefix `user1 /` to the bucket `app - base - oss`. However, the application cannot upload objects with other prefixes. Using this kind of policy, you can isolate applications with different prefixes in a bucket.

```

{
  "Statement ": [
    {
      "Action ": [
        " oss : PutObject "
      ],
      "Effect ": " Allow ",
      "Resource ": ["acs : oss :*: *: app - base - oss / user1 /
*", "acs : oss :*: *: app - base - oss "]
    }
  ],
  "Version ": " 1 "
}

```

Operations on OSS	Result
List all created buckets.	Failed

Operations on OSS	Result
Upload an object without a prefix, such as text.txt.	Failed
Download an object without a prefix, such as test.txt.	Failed
Upload an object with a prefix, such as user1/test.txt.	Success
Download an object with a prefix, such as user1/test.txt.	Failed
List objects without prefixes, such as test.txt.	Failed
List objects with prefixes, such as user1/test.txt.	Failed

- Read/write policies for all objects

A read/write policy for all objects indicates that an application can upload objects to the bucket `app - base - oss` and list, download, and delete all objects in the bucket.

```
{
  "Statement": [
    {
      "Action": [
        "oss : GetObject ",
        "oss : PutObject ",
        "oss : DeleteObject ",
        "oss : ListParts ",
        "oss : AbortMulti partUpload ",
        "oss : ListObject s "
      ],
      "Effect": "Allow",
      "Resource": ["acs : oss :*:*: app - base - oss /*", "acs : oss :*:*: app - base - oss "]
    }
  ],
  "Version": " 1 "
}
```

Operations on OSS	Result
List all created buckets.	Failed
Upload an object without a prefix, such as text.txt.	Success
Download an object without a prefix, such as text.txt.	Success

Operations on OSS	Result
Upload an object with a prefix, such as user1/test.txt.	Success
Download an object with a prefix, such as user1/test.txt.	Success
List objects without prefixes, such as test.txt.	Success
List objects with prefixes, such as user1/test.txt.	Success

- Read/write policies for objects with a specified prefix

This kind of policy indicates that an application can upload objects with the prefix `user1 /` to the bucket `app - base - oss` and list, download, and delete all objects with the prefix in the bucket. However, the application cannot perform read or write operations on objects with other prefixes. Using this kind of policy, you can isolate applications with different prefixes in a bucket.

```
{
  "Statement": [
    {
      "Action": [
        "oss : GetObject ",
        "oss : PutObject ",
        "oss : DeleteObject ",
        "oss : ListParts ",
        "oss : AbortMulti partUpload ",
        "oss : ListObject s "
      ],
      "Effect": "Allow ",
      "Resource": [
        "acs : oss :*:*: app - base - oss / user1 / *",
        "acs : oss :*:*: app - base - oss "
      ]
    }
  ],
  "Version": " 1 "
}
```

Operations on OSS	Result
List all created buckets.	Failed
Upload an object without a prefix, such as text.txt.	Failed
Download an object without a prefix, such as text.txt.	Failed
Upload an object with a prefix, such as user1/test.txt.	Success

Operations on OSS	Result
Download an object with a prefix, such as user1/test.txt.	Success
List objects without prefixes, such as test.txt.	Success
List objects with prefixes, such as user1 /test.txt.	Success

Complex policy examples

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": [
        "oss:GetBucketAcl",
        "oss:ListObjects"
      ],
      "Resource": [
        "acs:oss:*:1775305056:529849:mybucket"
      ],
      "Effect": "Allow",
      "Condition": {
        "StringEquals": {
          "acs:UserAgent": "java - sdk",
          "oss:Prefix": "foo"
        },
        "IpAddress": {
          "acs:SourceIp": "192.168.0.1"
        }
      }
    },
    {
      "Action": [
        "oss:PutObject",
        "oss:GetObject",
        "oss:DeleteObject"
      ],
      "Resource": [
        "acs:oss:*:1775305056:529849:mybucket/file"
      ],
      "Effect": "Allow",
      "Condition": {
        "IpAddress": {
          "acs:SourceIp": "192.168.0.1"
        }
      }
    }
  ]
}
```

The preceding example describes a complex authorization policy. By using this policy, a user can authorize other users through RAM or STS. In the policy, a statement is

included (a policy can include multiple statements), in which Action, Resource, Effect, and Condition are specified.

This policy grant permissions to authorized users so that they can access your resources, such as `mybucket` and `mybucket / file *`. In addition, this policy supports the following operations: `GetBucketAcl`, `GetBucket`, `PutObject`, `GetObject`, and `DeleteObject`. Conditions included in Condition indicate that authentication is successful and authorized users can access related resources only when `UserAgent` is `java-sdk` and the source IP address is `192.168.0.1`. The Prefix condition is used when the `GetBucket` (`ListObjects`) action is performed. For more information about the field, see OSS API documentation.

Version

The Version field specifies the version of the policy. For the configuration method in this document, it is set to `1`.

Statement

A statement describes the authorization semantics. According to different scenarios, a statement can include multiple semantics which include Action, Effect, Resource, and Condition individually. When receiving a request, the system checks all Statements in the policy. All Statements that match the request are classified into two categories based on their Effect settings: Allow or Deny, in which Deny statements have higher priority when the system determines whether the authentication is successful. If all matched statements are classified into Allow, the request passes the authentication. If a matched statement is classified into Deny, or no statement matches the request, the request is rejected.

Action

Actions can be classified into three categories:

- Service-level actions: include the `GetService` action used to list the buckets owned by a user.
- Bucket-level actions: indicate actions performed on buckets, such as `oss:PutBucketAcl` and `oss:GetBucketLocation`. The name of each action corresponds to an API.
- Object-level actions: indicate actions performed on objects, such as `oss:GetObject`, `oss:PutObject`, `oss>DeleteObject`, and `oss:AbortMultipartUpload`.

To authorize a type of actions on objects, you can select one or more of the preceding actions. In addition, all action names must be prefixed with `oss :`, as shown in the preceding example. The Action field is a list that can include multiple actions. The following tables show the mapping relationship between actions and APIs.

- Service-level actions

API	Action
GetService (ListBuckets)	oss:ListBuckets

- Bucket-level actions

API	Action
PutBucket	oss:PutBucket
GetBucket (ListObjects)	oss:ListObjects
PutBucketAcl	oss:PutBucketAcl
DeleteBucket	oss>DeleteBucket
GetBucketLocation	oss:GetBucketLocation
GetBucketAcl	oss:GetBucketAcl
GetBucketLogging	oss:GetBucketLogging
PutBucketLogging	oss:PutBucketLogging
DeleteBucketLogging	oss>DeleteBucketLogging
GetBucketWebsite	oss:GetBucketWebsite
PutBucketWebsite	oss:PutBucketWebsite
DeleteBucketWebsite	oss>DeleteBucketWebsite
GetBucketReferer	oss:GetBucketReferer
PutBucketReferer	oss:PutBucketReferer
GetBucketLifecycle	oss:GetBucketLifecycle
PutBucketLifecycle	oss:PutBucketLifecycle
DeleteBucketLifecycle	oss>DeleteBucketLifecycle
ListMultipartUploads	oss:ListMultipartUploads
PutBucketCors	oss:PutBucketCors
GetBucketCors	oss:GetBucketCors
DeleteBucketCors	oss>DeleteBucketCors

API	Action
PutBucketReplication	oss:PutBucketReplication
GetBucketReplication	oss:GetBucketReplication
DeleteBucketReplication	oss>DeleteBucketReplication
GetBucketReplicationLocation	oss:GetBucketReplicationLocation
GetBucketReplicationProgress	oss:GetBucketReplicationProgress

- Actions on objects

API	Action
GetObject	oss:GetObject
HeadObject	oss:GetObject
PutObject	oss:PutObject
PostObject	oss:PutObject
InitiateMultipartUpload	oss:PutObject
UploadPart	oss:PutObject
CompleteMultipart	oss:PutObject
DeleteObject	oss>DeleteObject
DeleteMultipleObjects	oss>DeleteObject
AbortMultipartUpload	oss:AbortMultipartUpload
ListParts	oss:ListParts
CopyObject	oss:GetObject,oss:PutObject
UploadPartCopy	oss:GetObject,oss:PutObject
AppendObject	oss:PutObject
GetObjectAcl	oss:GetObjectAcl
PutObjectAcl	oss:PutObjectAcl
RestoreObject	oss:RestoreObject

Resource

The resource field indicates specified resources or a kind of resources (which can be represented by a wildcard *). The format of a resource is as follows: `acs : oss`

`:{ region }:{ bucket_owner }:{ bucket_name }/{ object_name }`.

The "{object_name}" part is not required for the names of bucket-level actions. The

format of a resource for a bucket-level action is as follows: `acs : oss :{ region } :{ bucket_owner } :{ bucket_name }`. The Resource field is a list that can include multiple resources. The region field is not supported currently and is set to * in the preceding example.

Effect

The Effect field indicates the authorization result of this statement and has two values :Allow and Deny. When multiple statements match a request, statements in which the value of Effect is Deny has higher priority.

For example, the following policy prohibits users from deleting a specified directory but allows them to perform all operations on other objects.

```
{
  "Version": " 1 ",
  "Statement": [
    {
      "Effect": " Allow ",
      "Action": [
        " oss :*"
      ],
      "Resource": [
        " acs : oss :*:*: bucketname "
      ]
    },
    {
      "Effect": " Deny ",
      "Action": [
        " oss : DeleteObje ct "
      ],
      "Resource": [
        " acs : oss :*:*: bucketname / index /*",
      ]
    }
  ]
}
```

Condition

The Condition field indicates the conditions for the authorization policy. In the preceding example, you can set checking conditions for `acs:UserAgent` and `acs:SourceIp`, and use `oss:Prefix` as a condition to restrict resources when the `GetBucket` action is performed.

OSS supports the following conditions

Condition	Function	Valid value
<code>acs:SourceIp</code>	Specifies the source IP address or IP range.	IP address or IP range, wildcards (*) supported

Condition	Function	Valid value
acs:UserAgent	Specifies the http useragent header.	String
acs:CurrentTime	Specifies a valid access time.	Time in the ISO8601 format
acs:SecureTransport	Indicates whether the HTTPS protocol is used.	"true" or "false"
oss:Prefix	Indicates the prefix used when the ListObjects action is performed.	Valid object name

Best practice

OSS provides [Ram Policy Editor](#) that can help you generate a RAM policy quickly.

You can also [Grant permissions with a simple policy](#) by using ossbrowser, a graphical management tool to authorize a RAM user so that it can access specified buckets or directories.

For more examples of configuring authorization policies in different scenarios, see [Tutorial: control access to buckets and objects](#) and [Authorization for OSS](#).

14.4 Bucket policy

You can configure bucket policies to authorize users to access your OSS buckets. Compared with a RAM policy, bucket policies can be directly configured by the bucket owner on the console for access authorization.

Bucket policies are suitable for the following scenarios:

- Authorize RAM users of other accounts to access your OSS resources.

You can authorize RAM users of other accounts to access your OSS resources.

- Authorize anonymous users to access your OSS resources using specific IP addresses or IP ranges.

In some cases, you must authorize anonymous users to access OSS resources using specific IP addresses or IP ranges. For example, confidential documents of an enterprise are only allowed to be accessed within the enterprise but not in other regions. Previously, configuring RAM policies for every user was a tedious and complex task because of the potential for a large number of internal users. To

resolve this issue, you can configure access policies with IP restrictions based on bucket policies to authorize a large number of users easily and efficiently.

For more information about the configuration methods of bucket policies and video tutorials, see [Use bucket policies to authorize other users to access OSS resources](#).

14.5 Cross-account authorization

14.5.1 Overview

OSS provides multiple cross-account authorization methods to allow users using different accounts to access OSS resources.

The ACL for all OSS resources is private by default. The owner of a OSS resource can grant permissions to users using different accounts so that they can access the OSS resource. The following cross-account authorization methods can be used to allow other users to access OSS resources.

- [Authorize a RAM user under another Alibaba Cloud account by adding a bucket policy](#): A bucket policy authorize users based on resources. Compared with RAM policies, bucket policies can be easily configured in the graphical console. By configuring bucket policies, you can directly authorize other users so that they can access your bucket even you do not have permissions for RAM operations. You can configure bucket policies to grant bucket access permissions with IP address restrictions to anonymous users and RAM users under other accounts.
- [Tutorial: Grant cross-account bucket permissions](#): The RAM administrator can configure a RAM role and add the ID of another Alibaba Cloud account as trusted ID. After that, the RAM administrator can grant OSS access configuration permissions to the RAM role to share OSS resources to users under the Alibaba Cloud account.

14.5.2 Tutorial:Authorize a RAM user under another Alibaba Cloud account by adding a bucket policy

The ACL for an OSS resource is private by default. To allow another user to access your OSS resources, you can grant permissions for the user to access your bucket by adding a bucket policy.

For example: Company A wants its partner, company B, to access its OSS resources, but company A does not want to create a RAM user under its Alibaba Cloud account for this requirement. In this case, company A can grant permissions for company B

to access the bucket of company A by adding a bucket policy. After being authorized, company B can access an OSS resource owned by company A by adding the path of the resource in the OSS console.

Add a bucket policy for the RAM user of company B

- Follow these steps by using the Alibaba Cloud account of company B:
 1. Log on to the [RAM console](#) and create a RAM user. For more information, see [Create a RAM user](#).
 2. In the RAM console, click Users.
 3. Click the created RAM user and record its UID.
- Follow these steps using the Alibaba Cloud account of company A:
 1. Log on to the [OSS console](#).
 2. In the left-side bucket list, click the name of the bucket that you want to grant permissions for company B.
 3. Click Filled > Authorize > Authorize.
 4. In the Authorize dialog box, enter the policy information. Select Other Account for Accounts, and enter the UID of the RAM user created by company B. For more information about other parameters, see [Use bucket policies to authorize other users to access OSS resources](#).
 5. Click OK.

Log on to OSS with the RAM user of company B and add the resource path

After a bucket policy is added, you must log on to the OSS console with the RAM user of company B and add the access path of the OSS resource of company A. To add the access path, follow these steps:

1. Log on to Alibaba Cloud console with the RAM user of company B through the [RAM user logon link](#).
2. Open the [OSS console](#).

3. In the left-side menu, click "+" on the right of My OSS Paths. In the displayed Add Authorized OSS Path dialog box, add the following information:

- **Region:** Select the region of the bucket that company A allows company B to access.
- **OSS path:** Add the resource path that company A allows company B to access. The format of an OSS path is as follows: bucket/object-prefix. For example, if company A allows company B to access only the `abc` folder in the `aliyun` bucket, the OSS path is `aliyun / abc`.

You can also [Create an AccessKey](#) for the RAM user, and use [ossutil](#) or [ossbrowser](#) with the AccessKey to access the authorized bucket.

References

You can also grant permissions for other users to access your OSS resources in the following methods:

- [Tutorial: Authorize a RAM user under another Alibaba Cloud account by creating a RAM role](#)

15 Manage logs

15.1 Access logging

When you access OSS, a large number of access logs are generated. After you enable the logging feature for a bucket, OSS automatically generates an object by hour based on the predefined naming rules to store access logs for the bucket and writes the object to the specified bucket. You can use Alibaba Cloud Data Lake Analytics or build a Spark cluster to analyze access logs. You can also set lifecycle management rules for the bucket that stores access logs to convert the storage class of log objects to Archive for long-term archiving.

**Note:**

For more information about logging-related APIs, see the following topics:

- Enable the logging feature: [PutBucketLogging](#)
- Disable the logging feature: [DeleteBucketLogging](#)
- View the logging configuration: [GetBucketLogging](#)

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Node.js SDK	
Ruby SDK	

Naming rules for objects that store access logs

```
< TargetPref ix >< SourceBucket > YYYY - mm - DD - HH - MM - SS - UniqueString
```

In the naming rules:

- **TargetPref ix** : the name prefix of the object that stores access logs. This field is user-defined and can be left empty.
- **YYYY - mm - DD - HH - MM - SS** : the year, month, day, hour, minute, and second when the object was created. (Note the number of digits.)
- **UniqueString** : the string (UUID) generated by OSS, which is used to uniquely identify the object.

The following example shows the name of an object that stores OSS access logs:

```
MyLog - oss - example2017 - 09 - 10 - 04 - 00 - 00 - 0000
```

In the preceding example,

- **MyLog -** indicates the object prefix specified by the user.
- **oss - example** indicates the name of the source bucket.
- **2017 - 09 - 10 - 04 - 00 - 00** indicates the time the object was created.
- **0000** indicates the string generated by OSS to uniquely identify the object.

Log object format

The following table describes the fields that comprise a log object. In such an object, these fields are combined in order from left to right and are separated by spaces.

Name	Example	Description
Remote IP	119.xxx.xx.11	The IP address from which the request is initiated. (The proxy or user firewall may block this field.)
Reserved	None	The reserved field.
Reserved	None	The reserved field.
Time	[02/May/2012:00:00:04 +0800]	The time OSS receives the request.
Request-URI	"GET /aliyun-logo.png HTTP/1.1"	The URI of the user request, including query-string.

Name	Example	Description
HTTP Status	200	The HTTP status code returned by OSS.
SentBytes	5576	The amount of traffic downloaded by the user from OSS.
RequestTime (ms)	71	The duration used to complete the request, in milliseconds.
Referer	<code>http://www.aliyun.com/product/oss</code>	The HTTP referer of the request.
User-Agent	<code>curl/7.15.5</code>	The User-Agent field in the HTTP header.
HostName	<code>oss-example.oss-cn-hangzhou.aliyuncs.com</code>	The domain to be accessed.
Request ID	<code>505B016950xxxxxx032593A4</code>	The UUID used to uniquely identify the request.
LoggingFlag	<code>true</code>	Indicates whether the access logging feature is enabled.
Requester Aliyun ID	<code>16571xxxxxx83691</code>	The RAM user ID, which is a hyphen (-) for anonymous access.
Operation	<code>GetObject</code>	The request type.
Bucket	<code>oss-example</code>	The name of the bucket to be accessed.
Key	<code>/aliyun-logo.png</code>	The name (key) of the object that the user requests.
ObjectSize	5576	The object size.
Server Cost Time (ms)	17	The duration for the OSS server to process this request, in milliseconds.
Error Code	<code>NoSuchBucket</code>	The error code returned by OSS.
Request Length	302	The length of the user request, in bytes.

Name	Example	Description
UserID	16571xxxxxx83691	The ID of the bucket owner .
Delta DataSize	280	The bucket size variation, which is a hyphen (–) if the bucket size does not change.
Sync Request	None	Indicates whether the request is a CDN back-to-origin request. The value is a hyphen (–) if the request is not a back-to-origin request.
Reserved	None	The reserved field.

Detail analysis

- The source bucket and destination bucket can be the same bucket, or different buckets that are owned by the same Alibaba Cloud account and in the same region. You can also store the access logs of multiple source buckets in the same destination bucket. In this case, we recommend that you specify different TargetPrefix values for the log objects of different source buckets.
- OSS generates an object that stores bucket access logs on an hourly basis. However , requests in the last hour may be recorded in the object generated for the last hour or the next hour.
- Each time OSS generates an object that stores bucket access logs, it performs a PUT operation and records the storage space that the operation occupies. However, OSS does not record the traffic generated by the PUT operation. After a log object is generated, you can perform operations on it as a common object.
- OSS ignores all query-string parameters whose values are prefixed with `x -`. However, these parameters are recorded in access logs. To easily identify a special

request from a large number of access logs, you can add a query-string parameter whose value is prefixed with `x -` to the URL of the request. Example:

```
http :// oss - example . oss - cn - hangzhou . aliyuncs . com /  
aliyun - logo . png
```

```
http :// oss - example . oss - cn - hangzhou . aliyuncs . com /  
aliyun - logo . png ? x - user = admin
```

OSS returns the same result for the preceding two requests. However, you can search for access logs that contain `x - user = admin` to easily locate the marked request.

- A hyphen (-) may appear in any field in OSS logs. It indicates that data is unknown or the field is invalid for the current request.
- More fields will be added to the end of OSS logs in the future as needed. We recommend that developers consider potential compatibility issues when developing log processing tools.

16 Data encryption

16.1 Server-side encryption

This topic describes how to use the server-side encryption feature provided by OSS to encrypt and protect persistent data stored in OSS.

OSS supports server-side encryption for uploaded data. This means that when user data is uploaded, OSS encrypts the data and permanently stores the data. Then, when the data is downloaded by a user, OSS automatically decrypts the data, returns the original data to the user, and declares in the header of the returned HTTP request that the data has been encrypted on the server.

Scenarios

The following server-side encryption methods are available for different application scenarios:

- Server-side encryption that uses CMKs managed by KMS for encryption and decryption (SSE-KMS)

When uploading an object, you can use a specified CMK ID or the default CMK managed by KMS to encrypt and decrypt a large amount of data. This method is cost-effective because you do not need to send user data to the KMS service side through networks for encryption and decryption.



Note:

- Fees for API calls are incurred if you use a CMK to encrypt an object.

- Server-side encryption fully managed by OSS (SSE-OSS)

This encryption method is a property of an object. When sending a request to upload an object or modify the metadata of an object, you can include the `x-oss-server-side-encryption` header in the request and specify its value as AES256. In this method, OSS uses AES256 to encrypt each object with an individual key. Furthermore, the individual keys are encrypted by a customer master key (CMK) that is updated periodically for higher security. This method applies to encrypt or decrypt bulk data.

**Notice:**

Only one server-side encryption method can be used for an object at one time.

Configuration

For detailed information about server-side encryption configuration, see [Protect data by performing server-side encryption](#).

Server-side encryption that uses CMKs managed by KMS for encryption and decryption

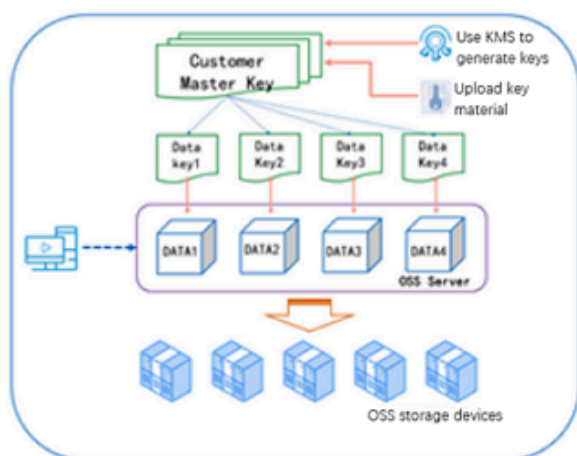
Key Management Service (KMS) is a secure and easy-to-use key protection and management service provided by Alibaba Cloud. KMS allows you to use keys in a secure manner, at minimal cost. You can view and manage keys in the KMS console.

To encrypt an object when creating it, you can include the `x-oss-server-side-encryption` header in the request and specify its value to `KMS` (which indicates that KMS is used for key management).

In addition to the usage of AES256 encryption algorithm, KMS stores the customer master key (CMK) used to encrypt data keys, generate data keys, and uses the envelope encryption mechanism to protect data from unauthorized access.

The following table shows the logic of SSE-KMS.

Server-side encryption (SSE-KMS): Supports using BYOK material for encryption



- Multi-level key envelope encryption
- Supports using BYOK material for encryption and decryption
- Keys are managed by KMS
- A random key is generated for each object

A CMK can be generated in the following methods:

- Use the default CMK managed by KMS.

When sending a request to upload an object or modify the metadata of an object, you can include the `x-oss-server-side-encryption` header in the request and specify its value as KMS without a specified CMK ID. In this method, OSS generates an individual key to encrypt each object by using the default managed CMK, and automatically decrypts the object when it is downloaded.

- Use a CMK specified by the user.

When sending a request to upload an object or modify the metadata of an object, you can include the `x-oss-server-side-encryption` header in the request, specify its value as KMS, and specify the value of `x-oss-server-side-encryption-key-id` to a specified CMK ID. In this method, OSS generates an individual key to encrypt each object by using the specified CMK, and adds the CMK ID used to encrypt an object into the metadata of the object so that the object is automatically decrypted when it is downloaded by an authorized user.

- Use the BYOK material of the user as the CMK.

You can import your BYOK material into KMS as the CMK as follows:

1. Create a CMK without key material.
2. Import the key material from an external source.

For more information about how to import key material, see [Import key material](#).



Note:

- If you use a CMK to encrypt an object, the data key used in the encryption is also encrypted and is stored as the metadata of the object.
- In server-side encryption that uses the default CMK managed by KMS, only the data in the object is encrypted. The metadata of the object is not encrypted.
- To use a RAM user to encrypt objects with a specified CMK, you must grant the relevant permissions to the RAM user. For more information, see [Use RAM for KMS resource authorization](#).

Server-side encryption fully managed by OSS

In this server-side encryption method, OSS generates and manages the keys used for data encryption, and provides strong multi-factor security measures to protect data

. AES256 (256-bit advanced encryption standard), a strong encryption algorithm, is used to encrypt data.

In this way, the encryption method becomes a property of an object. To perform server-side encryption on an object, you can include the `x-oss-server-side-encryption` header in the PutObject request and specify its value as `AES256`.

APIs that support server-side encryption



Notice:

This function is in the beta testing phase. To join the testing group, contact Alibaba Cloud technical support or open a ticket.

· APIs that support server-side encryption in requests

The `x-oss-server-side-encryption` header is supported in requests initiated by the following APIs:

- PutObject
- CopyObject
- InitiateMultipartUpload

The following table describes the HTTP headers that can be included in requests.

Header	Description	Example
x-oss-server-side-encryption	Specifies the server-side encryption method. Valid values: AES256 and KMS	<code>x-oss-server-side-encryption : KMS</code> indicates that the server-side encryption uses CMKs managed by KMS.
x-oss-server-side-encryption-key-id	Specifies the ID of the CMK used to encrypt the object. This header must be specified when you use a specified CMK ID for encryption.	<code>x-oss-server-side-encryption-key-id : 72779642-7d88-4a0f-8d1f-1081a9cc7a fb</code>



Note:

- If the `x-oss-server-side-encryption` header is included in requests initiated by APIs except for `PutObject`, `CopyObject`, and `InitiateMultipartUpload`, OSS returns HTTP status code 400 and includes `InvalidArgument` in the error message.
- If an invalid value is specified for the `x-oss-server-side-encryption` header, OSS returns HTTP status code 400 and includes `InvalidEncryptionAlgorithmError` in the error message.

- APIs that support server-side encryption in responses

OSS includes the `x-oss-server-side-encryption` header in responses to requests initiated by the following APIs to access objects encrypted at the server side.

- `PutObject`
- `CopyObject`
- `InitiateMultipartUpload`
- `UploadPart`
- `CompleteMultipartUpload`
- `GetObject`
- `HeadObject`

17 Static website hosting

17.1 Configure static website hosting

You can use the PutBucketWebsite API of OSS to set your bucket to the static website hosting mode and access the static website from the bucket endpoint.



Note:

For more information about the PutBucketWebsite API, see [PutBucketWebsite](#).

If your selected bucket is located in China (Hangzhou), after the configuration takes effect, the domain of the static website is as follows:

```
http ://< Bucket >. oss - cn - hangzhou . aliyuncs . com /
```



Note:

When you use the default endpoint to access a Webpage object in OSS in Mainland China or Hong Kong through the Internet, `Content - Dispositio n :`
`attachment = filename ;'` is automatically added to the response header. That is, when you use a browser to access a Webpage object, the object is downloaded as an attachment. If you use a custom domain to access OSS, the information is not added to the response header. For more information about how to use a custom domain to access OSS, see [Bind a custom domain](#).

OSS provides the following features to help you manage static websites hosted in OSS more easily:

- Index document support

The index document is the default index page (such as index.html) that OSS returns when a user directly accesses the root domain of a static website. If you set a bucket to the static website hosting mode, you must specify the index document.

- Error document support

The error document is an error page that OSS returns if an HTTP 4XX error (such as 404 NOT FOUND) occurs when a user accesses a static website. By specifying the error document, you can provide your users with appropriate error messages.

For example, you set the index document to `index.html`, the error document to `error.html`, the bucket name to `oss-sample`, and the endpoint to `oss-cn-hangzhou.aliyuncs.com`.

- When a user accesses `http://oss-sample.oss-cn-hangzhou.aliyuncs.com/` and `http://oss-sample.oss-cn-hangzhou.aliyuncs.com/directory/`, the user actually accesses `http://oss-sample.oss-cn-hangzhou.aliyuncs.com/index.html`.
- If the object does not exist when a user accesses `http://oss-sample.oss-cn-hangzhou.aliyuncs.com/object`, OSS returns `http://oss-sample.oss-cn-hangzhou.aliyuncs.com/error.html`.

Operating methods

Operating method	Description
Console	Web application, which is intuitive and easy to use
Java SDK	SDK demos in various languages
Python SDK	
PHP SDK	
Go SDK	
C SDK	
.NET SDK	
Node.js SDK	
Ruby SDK	

Detail analysis

- On a static website, all Webpages are composed of static content, including scripts such as JavaScript that are run on the client. OSS does not support content that needs to be processed by the server, such as PHP, JSP, and ASP.NET content.
- To access a bucket-based static website by using a custom domain, you can [bind a custom domain](#).

- When you set a bucket to the static website hosting mode:
 - The index document is required and the error document is optional.
 - The specified index document and error document must be objects in the bucket.

**Note:**

If you use an Archive bucket, you must specify Standard objects or restored Archive objects as the index document and error document. Otherwise, the static website cannot be accessed.

- After you set a bucket to the static website hosting mode:
 - OSS returns the index document for anonymous access to the root domain of the static website, and returns the result of the GetBucket operation for authorized access to the root domain of the static website.
 - OSS returns a specified object to users who access the root domain of a static website or an object that does not exist in OSS. OSS also charges a fee for such requests and the generated traffic.

Reference

- [Tutorial: Host a static website using a custom domain name](#)
- [Tutorial: Configure static website hosting](#)

17.2 Tutorial: Host a static website using a custom domain name

Suppose that you want to host a static website on Alibaba Cloud Object Storage Service (OSS). You have registered a domain (for example, `examplewebsite.com`), and you want the requests for `http://examplewebsite.com` and `http://www.examplewebsite.com` to be serviced from your OSS content. Whether you have an existing static website that you want to host on OSS, or you are starting from scratch, you can use this example and learn how to host websites on Alibaba Cloud OSS.

Prerequisites

This tutorial covers the following services:

- Domain name registration

If you do not have a registered domain name, such as `exampleweb site . com`, select a registrar to register one. Alibaba Cloud also provides domain name registration service. For more information, see [Alibaba Cloud Domain service](#).

- Alibaba Cloud OSS

You use Alibaba Cloud OSS to create buckets, upload a sample website page, configure permissions to let others see the content, and then configure the buckets for website hosting. In this example, because you want to allow requests for `http :// exampleweb site . com` and `http :// www . exampleweb site . com`, you create two buckets. You host content in only one bucket, and configure the other bucket to redirect requests to the bucket that hosts the content.

- Alibaba Cloud DNS

As your Domain Name System (DNS) provider, you configure Alibaba Cloud DNS . In this example, you add your domain name to Alibaba Cloud DNS and define a CNAME record so that you can use your domain name instead of the OSS assigned access domain name to access your OSS buckets.

In this example, we use Alibaba Cloud DNS. We recommend that you use Alibaba Cloud DNS. However, you can use various registrars to define a CNAME record that points to an OSS bucket.



Note:

This tutorial uses `exampleweb site . com` as a domain name. Replace this domain name with the one that you have registered.

Step 1: Register a domain

If you already have a registered domain, you can skip this step. If you have never hosted a website, your first step is to register a domain, such as `exampleweb site . com`. You can use Alibaba Cloud Domain service to register a domain.

For more information, see [Buy a domain name](#) in the Alibaba Cloud Domain Quick Start.

Step 2: Create and configure buckets and upload data

You create two buckets to support requests from both the root domain such as

exampleweb site . com and subdomain such as http :// www . exampleweb

site . com . One bucket is used to store the content, and the other bucket is used to

redirect requests to the bucket that stores the content.

Step 2.1: Create two buckets

In this step, you log on to Alibaba Cloud OSS console with your Alibaba Cloud account credentials and create the following two buckets:

- **originbucket:** to store the content
- **redirectbucket:** to redirect requests to originbucket

1. Log on to the [OSS console](#).

2. Create two buckets, for example, originbucket and redirectbucket, one to store the content, and the other to redirect requests to the bucket that stores the content. Set the access control list (ACL) of the two buckets to Public Read so that everyone can see the content of the buckets.

For detailed procedures, see [Create a bucket](#).

3. Make yourself a note of the access domain name of the originbucket and redirectbucket. You will use them in later steps. You can find the access domain name of a bucket on the Overview tab page of the bucket, as shown in the following figure.

The screenshot shows the 'originbucket' Overview page in the OSS console. The 'Access Domain Name' section is highlighted, showing the endpoint 'originbucket.oss-cn-beijing.aliyuncs.com' for Internet Access.

	Endpoint ②	Access Domain Name ②	HTTPS
Internet Access ③	oss-cn-beijing.aliyuncs.com	originbucket.oss-cn-beijing.aliyuncs.com	Supported
ECS Address for Classic Network Access (Intranet) ④	oss-cn-beijing-internal.aliyuncs.com	originbucket.oss-cn-beijing-internal.aliyuncs.com	Supported
ECS Address for VPC Network Access (Intranet) ④	oss-cn-beijing-internal.aliyuncs.com	originbucket.oss-cn-beijing-internal.aliyuncs.com	Supported

4. Upload your website data to originbucket.

You will host your content out of the root domain bucket originbucket, and you will redirect requests for the subdomain bucket redirectbucket to the root domain bucket originbucket. You can store content in either bucket.

For this example, you host content in the originbucket bucket. The content can be any type of files, such as text files, photos, and videos. If you have not yet created a website, then you only need two files for this example. One file is used as the homepage of the website, and the other file is used as the error page of the website.

For example, you can create one file named index.html using the following HTML and upload it to the bucket. In a later step, you use this file name as the default homepage for your website.

```
< html >
  < head >
    < title > Hello  OSS ! </ title >
    < meta  charset =" utf - 8 ">
  </ head >
  < body >
    < p > Now  host  on  Alibaba  Cloud  OSS </ p >
    < p > This  is  the  index  page </ p >
  </ body >
</ html >
```

You create another file named error.html using the following HTML and upload it to the bucket. This file is used as the 404 error page of a website. In a later step, you use this file name as the default 404 page for your website.

```
< html >
< head >
  < title > Hello  OSS ! </ title >
  < meta  charset =" utf - 8 ">
</ head >
< body >
  < p > This  is  the  404  error  page </ p >
</ body >
</ html >
```

Step 2.2: Configure buckets for website hosting

When you configure a bucket for website hosting, you can access the website using the OSS assigned access domain name.

In this step, you configure originbucket as a website.

1. Log on to the [OSS console](#).
2. From the bucket name list, select originbucket .

3. Click the Basic Settings tab and find the `Static Page` area.
4. Click Edit, and then enter the following information:
 - **Default Homepage:** The index page (equivalent to `index.html` of the website). Only HTML files that have been stored in the bucket can be used. For this example, enter `index . html`.
 - **Default 404 Page:** The default 404 page returned when an incorrect path is accessed. Only HTML and image files that have been stored in the bucket can be used. If this field is left empty, the default 404 page is disabled. For this example, enter `error . html`.
5. Click Save.

Step 2.3: Configure the index page for redirect

Now that you have configured the default homepage and error page of the originbucket, you also need to configure the default homepage of redirectbucket.

To configure the index page for redirect, follow these steps:

1. Log on to the [OSS console](#).
2. From the bucket name list, select `redirectbucket`.
3. Click the Basic Settings tab and find the `Static Page` area.
4. Click Edit, and then enter `index . html` in the `Default Homepage` text box.
5. Click Save.

Step 3: Bind your domain name to your OSS buckets

Now that you have your root domain `exampleweb site . com` and your OSS bucket originbucket, bind your domain to your OSS buckets so that you can access the OSS buckets using your own domain name instead of the domain name assigned by OSS.

In this example, before you bind your domain `exampleweb site . com` to your OSS bucket originbucket, you have to use the OSS assigned domain name `originbucket.oss-cn-beijing.aliyuncs.com` to access your bucket originbucket. After you bind your domain `exampleweb site . com`, you can use this `exampleweb site . com` to access your OSS bucket.

Similarly, you also need to bind your subdomain `www . exampleweb site . com` to your OSS bucket redirectbucket, so that you can use `www . exampleweb`

`site . com` instead of the OSS assigned domain name `originbucket.oss-cn-beijing.aliyuncs.com` to access your OSS bucket.

To bind your root domain `exampleweb site . com` to your OSS bucket `originbucket`, follow these steps:

1. Log on to the [OSS console](#).
2. From the bucket name list, select `originbucket`.
3. Click the Domain Names tab.
4. Click Bind User Domain to open the Bind User Domain dialog box.
5. In the User Domain text box, enter the root domain `examplewebsite.com`.
6. Define a CNAME record to `originbucket`.
 - If your domain name has been resolved under your Alibaba Cloud account, you can open the Add CNAME Record Automatically switch. Then click Submit.
 - If your domain name has not been resolved under your Alibaba Cloud primary account, the Add CNAME Record Automatically switch is disabled. Follow these steps to add a CNAME record manually, and then click Submit.
 - a. Add your domain name in Alibaba Cloud DNS.
 - If your domain name is registered with Alibaba Cloud, it is automatically added to the Alibaba Cloud DNS list. You can skip this step.
 - b. In the Alibaba Cloud DNS console, find your domain name.
 - c. Click the domain name or click the Configure link.
 - d. Click Add Record.
 - e. In the Add Record dialog box, select CNAME from the Type drop-down box, and enter the OSS domain name of the bucket in the Value text box. In this example, enter `originbucket.oss-cn-beijing.aliyuncs.com`.
 - f. Click Confirm.
7. Follow the preceding steps to bind your sub domain `www . exampleweb site . com` to your OSS bucket `redirectbucket`.

Step 4: Configure your website redirect

Now that you have configured your bucket for website hosting and bound your custom domain to your OSS bucket, configure the `redirectbucket` to redirect

all requests for `http://www.examplewebsite.com` to `http://examplewebsite.com`.

To configure your website redirect, follow these steps:

1. Log on to the [OSS console](#).
2. From the bucket name list, select `redirectbucket`.
3. Click the Basic Settings tab and find the `Back-to-Origin` area.
4. Click Edit, and then click Create Rule.
5. Create the 404 redirect rule as follows:
 - a. In the Back-to-Origin Type area, select `Redirect`.
 - b. In the `Back-to-Origin When` area, set `HTTP Status Code` to 404.
 - c. In the `Back-to-Origin URL` area, select `Add a prefix or suffix`, enter your domain name of the originbucket. In this example, enter `examplewebsite.com`.
 - d. Click OK.

Step 5: (Optional) Speed up your website with Alibaba Cloud CDN

You can use Alibaba Cloud Content Delivery Network (CDN) to improve the performance of your website. CDN makes your website files (such as HTML, images, and video) available from data centers around the world. These are called edge nodes. When a visitor requests a file from your website, Alibaba Cloud CDN automatically redirects the request to a copy of the file at the nearest edge node. This results in faster download times than if the visitor had requested the content from a data center that is located farther away.

Alibaba Cloud CDN caches content at edge nodes for a period of time that you specify. If a visitor requests content that has been cached for longer than the expiration date, CDN checks the origin server to see if a later version of the content is available. If a later version is available, CDN copies the new version to the edge node. Changes that you make to the original content are replicated to edge nodes as visitors request the content. However, before the expiration date, the content is still in the earlier version. We recommend that you open the Auto Refresh CDN Cache switch, so that changes you make to the original content are automatically refreshed in CDN cache in real time.

To speed up originbucket with CDN, follow these steps:

1. Add a CDN domain in the CDN console. For detailed procedures, see Step 2. Add a CDN domain in [CDN quick start](#).
2. Define a CNAME record in the Alibaba Cloud DNS console. For detailed procedures, see [Configure Alibaba Cloud DNS](#).
3. Open the Auto Refresh CDN Cache switch in the OSS console.
 - a. Log on to the [OSS console](#).
 - b. From the bucket name list, select `originbucket` et .
 - c. Click the Domain Names tab.
 - d. Open the Auto Refresh CDN Cache switch.
4. Follow the preceding steps to speed up `redirectbucket` with CDN.

Step 6: Test the website

To verify that the website is working correctly, in your browser, try the following URLs:

URL	Result
<code>http://examplewebsite.com</code>	Displays the index document in the originbucket.
The URL of a file that does not exist in the originbucket, for example, <code>http://examplewebsite.com/abc</code>	Displays the error document in the originbucket.
<code>http://www.examplewebsite.com</code>	Redirects your request to <code>http://examplewebsite.com</code> .
<code>http://www.examplewebsite.com/abc</code>	Redirects your request to <code>http://examplewebsite.com/abc</code> .



Note:

In some cases, you may need to clear the cache of your web browser to see the expected behavior.

Cleanup

If you created your static website as a learning exercise only, remember to delete the Alibaba Cloud resources that you allocated to avoid unnecessary fees charged to your account. After you delete your Alibaba Cloud resources, your website is no longer available.

To clean up, follow these steps:

1. Disable and then remove the domain name in the Alibaba Cloud CDN console.
2. Delete the CNAME records in the Alibaba Cloud DNS console.
3. Delete the OSS files and buckets in the Alibaba Cloud OSS console.

18 OSS sandbox

When your OSS bucket is under attack or is used to distribute illegal content, OSS automatically adds the bucket to the sandbox. The bucket in the sandbox can still respond to requests. However, the service quality is degraded, which your application may be aware of.

- If your bucket is under attack, OSS automatically adds the attacked bucket to the sandbox. In this case, you must bear the cost incurred by the attack.
- If your user uses your bucket to distribute illegal content that involves pornography, politics, and terrorism, OSS also adds the bucket to the sandbox. Users will be held liable for serious violations.

**Note:**

After your bucket is added to the sandbox, you will receive SMS and email notifications.

Preventive measures against attacks

To prevent your bucket from being added to the sandbox due to attacks, you can use Anti-DDoS Pro to prevent DDoS attacks and HTTP floods. If your business is subject to attacks, you can use either of the following solutions to add preventive measures:

- **Solution 1: Attach a domain to the bucket and configure an Anti-DDoS Pro instance**
 1. Attach a custom domain to the bucket. For more information, see [Attach a custom domain](#).
 2. Purchase appropriate [Anti-DDoS Pro](#) based on your business needs.
 3. Attach the Anti-DDoS Pro instance to the custom domain you have set.

Domain Name:

Note: If a wildcard domain is added, please also add its top-level domain in another type. For example, after you add the *.taobao.com wildcard domain, you must add its top-level domain, taobao.com, in another rule. The top-level domain and sub-level domain must be configured separately.

Protocol: ☒ HTTP ☐ HTTPS ☐ websocket ☐ websockets

Origin IP/Domain: ☐ Origin Site IP ☒ Origin site domain

Domain:

If your source IP was exposed, please see [What to do after source IP exposed?](#)

[Next](#)

- **Domain Name:** Specify the custom domain you want to attach the Anti-DDoS Pro instance to.
- **Protocol:** Select one as needed.
- **Origin IP/Domain:** Click Origin site domain. Enter the default domain of your OSS.

For more information about the configurations of other parameters, see [Configure an Anti-DDoS Pro instance](#).

- **Solution 2: Configure a reverse proxy by using ECS and configure an Anti-DDoS Pro instance**

For security reasons, the IP address resolved from the default domain of a bucket changes dynamically. If you want to use a fixed IP address to access the bucket, you

can use ECS to set up a reverse proxy. You can attach the EIP of an ECS instance to an Anti-DDoS Pro instance to prevent DDoS attacks and HTTP floods. You can set up the reverse proxy as follows:

1. Create an ECS instance in CentOS or Ubuntu. For more information, see [Create an ECS instance](#).



Notice:

If the bucket encounters sporadic bursts of network traffic or access requests, upgrade hardware configurations of ECS or set up ECS clusters.

2. For more information about how to use ECS to configure a reverse proxy for access to OSS, see [Configure a reverse proxy for access to OSS](#).
3. Purchase appropriate [Anti-DDoS Pro](#) based on your business needs.
4. Attach the Anti-DDoS Pro instance to the domain you have set, as shown in [Step 3](#). Set Domain Name to the domain you want to attach to the EIP of the ECS instance and Origin IP/Domain to Origin Site IP. Enter the EIP of the ECS instance in the text box.

Solution comparison and analysis

Solution	Advantage	Disadvantage
Solution 1 : Bind a domain to the bucket and configure an Anti - DDoS Pro instance	Simple configurations : You can perform the configurations through the GUI in the console.	Scenarios: Only buckets that are not added to the sandbox can be protected.
Solution 2 : Configure a reverse proxy by using ECS and configure an Anti - DDoS Pro instance	<ul style="list-style-type: none"> • Application scope: All buckets can be protected, regardless of whether they are added to the sandbox. • Scenarios: Use a fixed IP address to access OSS. 	<ul style="list-style-type: none"> • Complex configurations : You must set up an NGINX reverse proxy. • High costs: You need to purchase ECS to set up an NGINX reverse proxy.

Preventive measures against operations that involve illegal content

To prevent your bucket being added to the sandbox due to the distribution of illegal content that involves pornography, politics, and terrorism, we recommend that you

activate [Content Moderation](#). Alibaba Cloud Content Moderation periodically scans your bucket to effectively monitor the distribution of illegal content.

To prevent your bucket being added to the sandbox due to the distribution of illegal content that involves pornography, politics, and terrorism, we recommend that you activate Content Moderation. Alibaba Cloud Content Moderation periodically scans your bucket to effectively monitor the distribution of illegal content.

What shall I do if a bucket is added to the sandbox

Alibaba Cloud does not provide migration services for buckets that are added to the sandbox. If your bucket is added to the sandbox, perform different operations based on different scenarios:

- A bucket is added to the sandbox due to attacks
 - If a bucket is added to the sandbox, configure security preventive measures, as shown in [Solution 2: Configure a reverse proxy by using ECS and configure an Anti-DDoS Pro instance](#).



Notice:

Create an ECS instance that is in the same region as your bucket. Set proxy_pass to the intranet domain of the bucket.

- If the bucket in your account encounters multiple attacks, buckets created after that bucket are also automatically added to the sandbox. To address this issue, follow these steps:
 1. Purchase Anti-DDoS Pro.
 2. Use the ticket system to submit the application of Created Bucket Not added to the Sandbox.
 3. After the application is approved, configure the Anti-DDoS Pro instance, as shown in [Solution 1: Attach a domain to the bucket and configure an Anti-DDoS Pro instance](#).

- A bucket is added to the sandbox due to the distribution of illegal content
 - If a bucket is added to the sandbox, follow these steps:
 1. Activate [Content Moderation](#). Alibaba Cloud Content Moderation periodically scans your bucket to prevent the distribution of illegal content.
 2. Use ECS to configure the reverse proxy, as shown in [Solution 2: Configure a reverse proxy by using ECS and configure an Anti-DDoS Pro instance](#).

**Notice:**

Create an ECS instance that is in the same region as your bucket. Set proxy_pass to the intranet domain of the bucket.

- If you own buckets that distribute illegal content at one time or over multiple times, these buckets and newly created buckets are automatically added to the sandbox. To address this issue, follow these steps:
 1. Purchase Content Moderation.
 2. Use the ticket system to submit the application of Created Bucket Not added to the sandbox.
 3. After the application is approved, configure Content Moderation to periodically detect illegal content distributed through your buckets.

19 Monitoring service

19.1 Monitoring service overview

The OSS monitoring service details metric data, including basic system operation statuses, performance, and metering. It also provides a custom alarm service to track requests, analyze usage, collect statistics on business trends, and promptly discover and diagnose system problems.

OSS metric indicators are classified into indicator groups such as basic service indicators, performance indicators, and metering indicators. For more information, see [Monitoring indicators reference](#).

High real-time performance

Real-time performance monitoring can expose potential peak/valley problems, display actual fluctuations, and provide insights into the analysis and evaluation of business scenarios. OSS real-time metric indicators (excluding the metering indicator) enable minute-level collection and aggregation of metric data with an output delay of less than one minute.

Metering indicator description

The metering indicator uses the following features:

- Metering entries are collected, aggregated, and output at the hour-level.
- However, the output delay can be up to thirty minutes.
- The time of metering refers to the start time of the relevant statistical period.
- The metering acquisition cutoff time is the end time of the last metering data statistical period for the current month. If no metering data is produced during the current month, the metering data acquisition cutoff time is 00:00 on the first day of the current month.
- A maximum amount of metering entries is pushed for presentation. For precise metering data, go to Billing Management and click [Usage Records](#).

For example, suppose that the user just uses the request to upload data, an average of 10 times a minute. So at 2016-05-10 08:00:00 to 2016-05-10 09:00:00 this hour of time, the measured data value of the user's number of put class requests is 600 times (10

*60 minutes), data time 2016-05-10 At 08:00:00, the data will be output at about 09:30:00. If this data is from 2016-05-01 From 00:00:00 to the present, the last measure monitoring data, then the cut-off time of the month for the measure data acquisition is 2016-05-10 09:00:00. If the user did not produce any measurement data in May 2016 , the cut off time for the measurement collection is 2016-05-01 00:00:00.

OSS alarm service

You can set up to 1,000 alarm rules.

Alarm rules can be configured for other metric indicators, which can then be added to alarm monitoring. Additionally, multiple alarm rules may be configured for a single metric indicator.

- For information about the alarm service, see [Alarm service overview](#).
- For instructions about how to use the OSS alarm service, see [Alarm service user guide](#).
- For more information about OSS metric indicators, see [Monitoring indicators reference](#).

Metric data retention policy

Metric data is retained for 31 days and is automatically cleared upon expiration.

To analyze metric data offline, or download and store historical metric data for longer than 31 days, use the appropriate tool or input code to read the data storage of CloudMonitor. For more information, see [Metric data access through the API](#).

The console displays metric data up until the past seven days. To view historical metric data earlier than seven days, use the CloudMonitor SDK. For more information, see [Metric data access through the API](#).

Metric data access through the API

The API of CloudMonitor allows you to access OSS metric data. For usage information , see:

- [CloudMonitor API Reference](#)
- [Cloud monitoring SDK Reference](#)
- [Metric item reference](#)

Monitoring, diagnosis, and troubleshooting

The following documentation provides monitoring, diagnosis, and troubleshooting details related to OSS management:

- [Real-time service monitoring](#)

Describes how to use the monitoring service to monitor the running status and performance of OSS.

- [Tracking and diagnosis](#)

Describes how to use the OSS monitoring service and logging function to diagnose problems, and how to associate relevant information in log files for tracking and diagnosis.

- [Troubleshooting](#)

Describes typical problems and corresponding troubleshooting methods.

Considerations

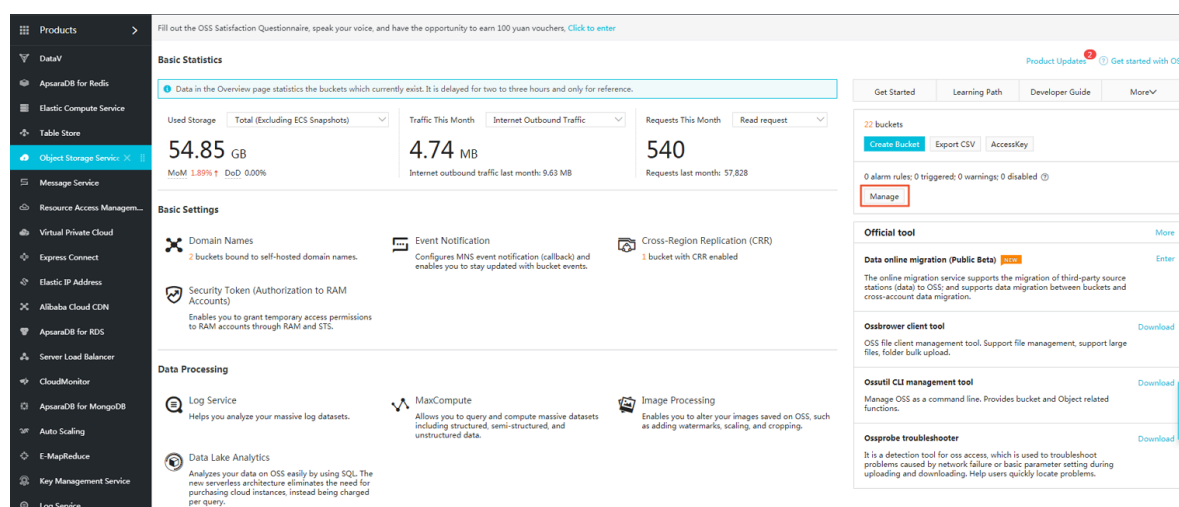
OSS buckets must be globally unique. If, after deleting a bucket, you create another bucket with the same name as the deleted bucket, the monitoring and alarm rules set for the deleted bucket are applied to the new bucket.

19.2 Monitoring service user guide

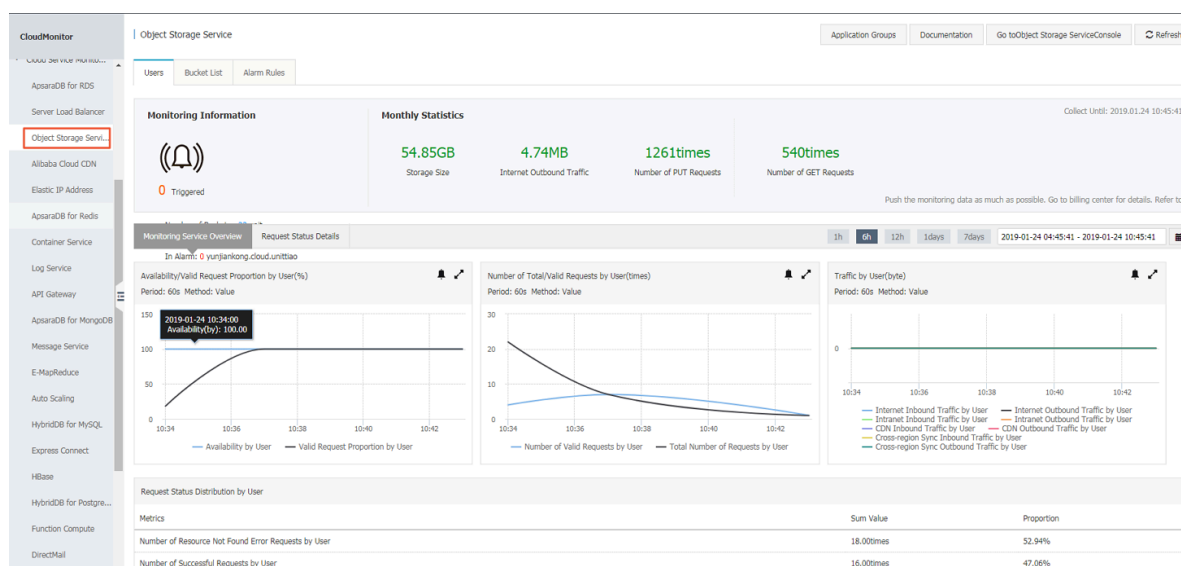
OSS monitoring entry

The OSS monitoring service is available on the Alibaba Cloud Console. You can access the OSS monitoring service in either of the following ways:

- Log on to the [OSS console](#) and then click **Manage** on the right side of OSS overview page.



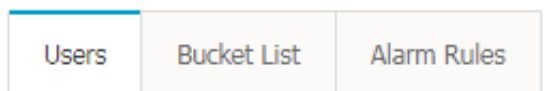
- Log on to the [CloudMonitor console](#) to view the OSS monitoring service.



OSS monitoring page

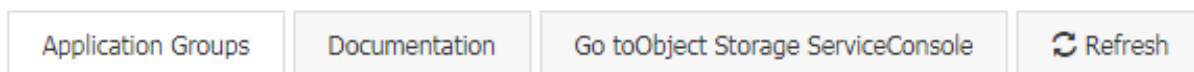
The OSS monitoring page consists of the following three tabs:

- Users
- Bucket list
- Alarm rules



The OSS monitoring page does not support automatic refresh. You can click Refresh in the upper-right corner to display the latest data.

Click Go to Object Storage Service Console to log on to the OSS console.

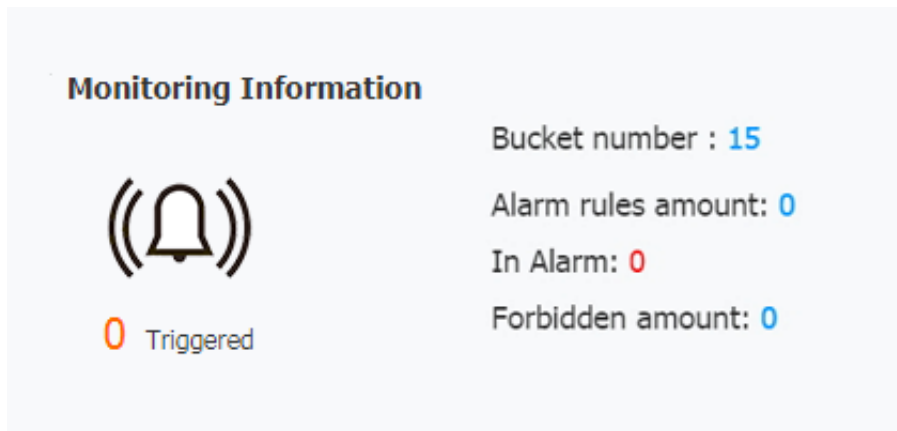


Users

The Users page displays monitoring information at the user level, including User monitoring information, Latest month statistics and User-level metric indicators.

- User monitoring information

This module shows the total number of your buckets and related alarm rules.

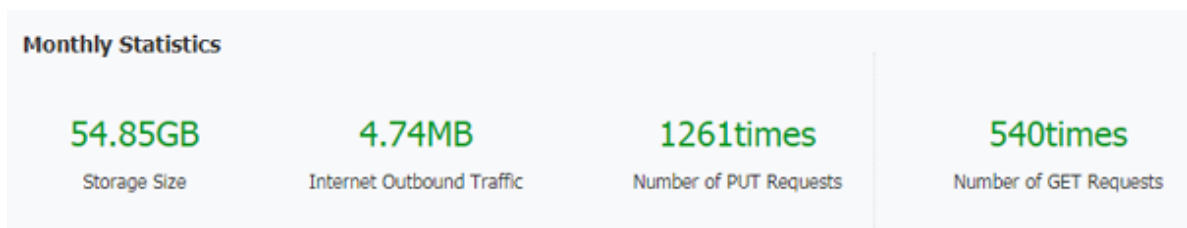


- ■ The parameters are as follows: Click the number next to Bucket number to display all the buckets you have created.
- Click the number next to Alarm rules amount, In Alarm, Forbidden amount, or Alerted to display the following information: Alarm rules amount refers to total number of alarm rules you have set.
- In Alarm refers to alarms in alarm state.
- Forbidden amount refers to alarms that are currently disabled.
- Alerted refers to alarms recently changed to alarm state

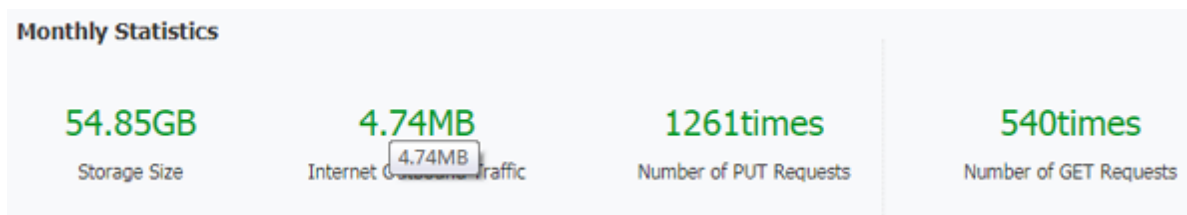
- Monthly Statistics

This module displays information about charged OSS resources that you have used during the period from 00:00 on the first day of the current month, to the metering acquisition cutoff time. The following indicators are displayed:

- Storage Size
- Internet Outbound Traffic
- Number of PUT Requests
- Number of GET Requests

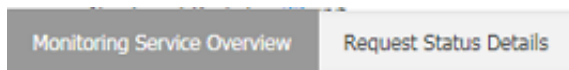


The unit of each value is automatically adjusted by the order of magnitude. The exact value is displayed when you hover the cursor over the selected value.



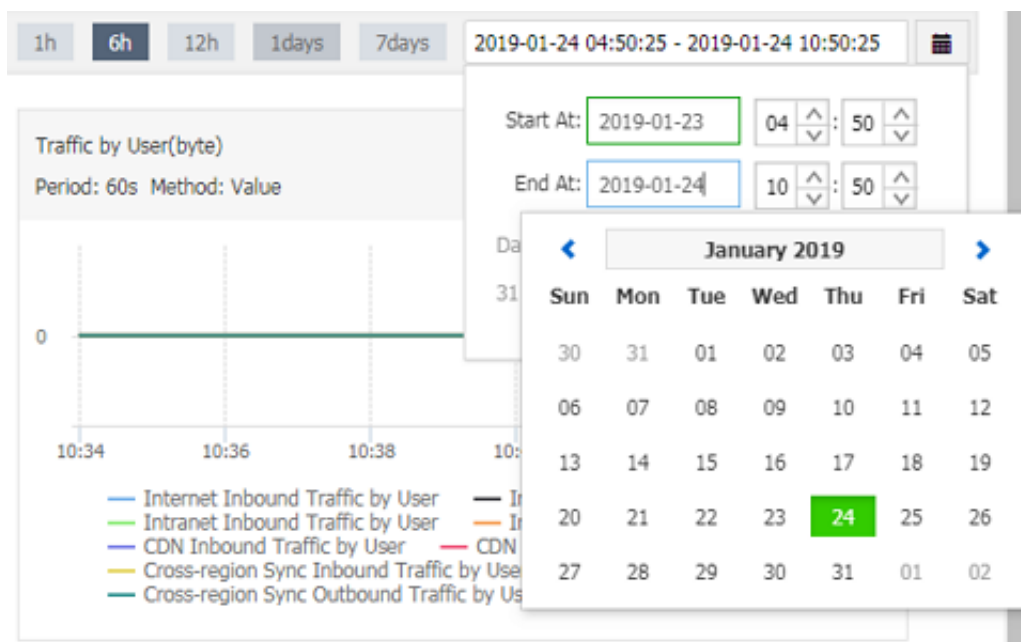
- User-level metric indicators

This module displays user-level metric charts and tables and consists of Monitoring Service Overview and Request Status Details .



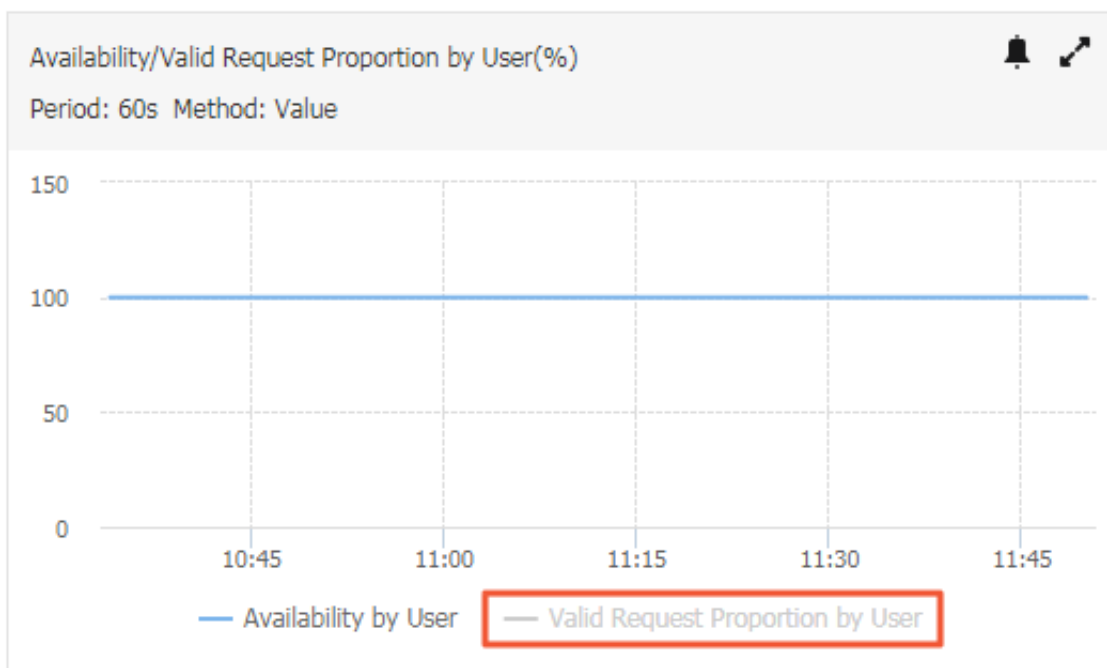
You can select a pre-determined time range, or define a time range in the custom time boxes, to display the corresponding metric chart or table.



- The following time range options are available: 1 hour, 6 hours, 12 hours, 1 day, and 7 days. The default option is 1 hour.
- The custom time boxes allow the start time and the end time to be defined at the minute-level.



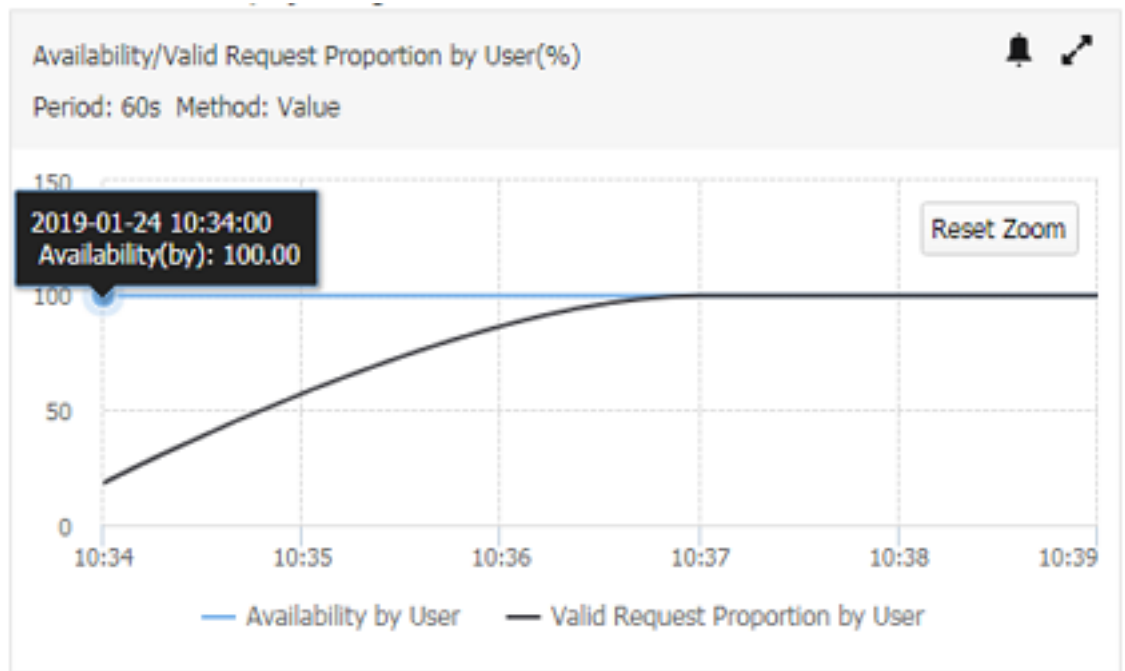
Metric charts/tables support the following display modes:

- Legend hiding: You can click a legend to hide the corresponding indicator curve, as shown in the following figure:



- Click the  icon in the upper-right corner of a metric chart to zoom in on the chart. Be note that tables cannot be zoomed in.
- Click the  icon in the upper-right corner of a metric chart to configure alarm rules for the displayed metric indicators. For more information, see the [Alarm Service User Guide](#). Be note that you cannot set alarm rules for tables and metering reference indicators.
- Place the cursor inside the curve area of a chart, and press and hold the left button on the mouse while dragging the mouse to

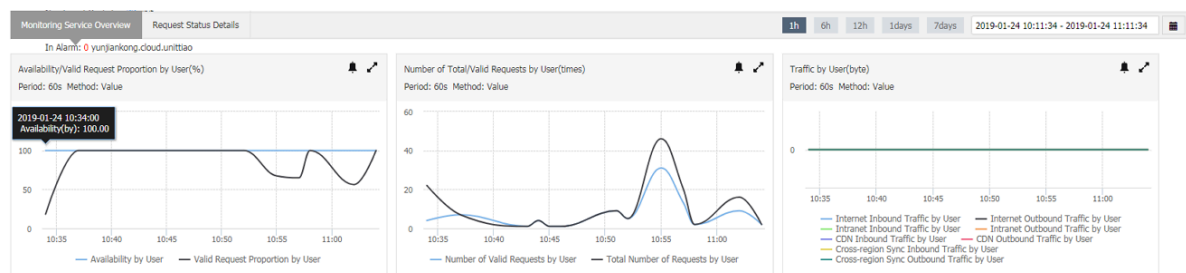
extend the time range. Click Reset Zoom to restore the original time range.



• Monitoring Service Overview

The Monitoring Service Overview page displays the following main metric charts:

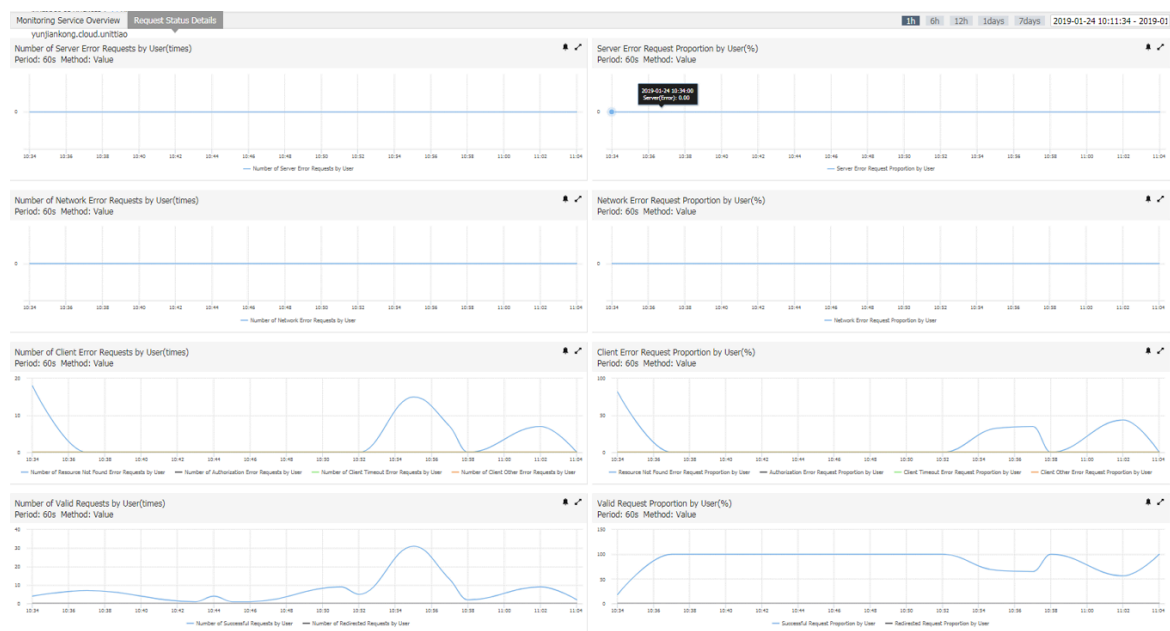
- User-level availability/valid request rate, which includes two metric indicators: availability and percentage of valid requests.
- User-level requests/valid requests, which includes two metric indicators: total number of requests and number of valid requests.
- User-level traffic, which includes eight metric indicators: Internet outbound traffic, Internet inbound traffic, Intranet outbound traffic, Intranet inbound traffic, CDN outbound traffic, CDN inbound traffic, outbound traffic of cross-region replication, and inbound traffic of cross-region replication.
- User-level request state distribution, which is a table that displays the number and percentage of each type of requests within the selected time range.



Request Status Details

The "Request State Details" page shows the metric data of request state distribution through the following main metric charts:

- Number of Server Error Requests by User
- Server Error Request Proportion by User
- Number of Network Error Requests by User
- Network Error Request Proportion by User
- Number of Client Error Requests by User, which includes four metric indicators : number of error requests indicating resource not found, number of authorization error requests, number of client-site time-out error requests, and number of other client-site error requests
- Client Error Request Proportion by User, which includes four metric indicators : percentage of error requests indicating resource not found, percentage of authorization error requests, percentage of client-site time-out error requests, and percentage of other client-site error requests
- Number of Valid Requests by User, which includes two metric indicators: number of successful requests and number of redirect requests
- Valid Request Proportion by User, which includes two metric indicators: percentage of successful requests and percentage of redirect requests



Bucket List

• Bucket list information

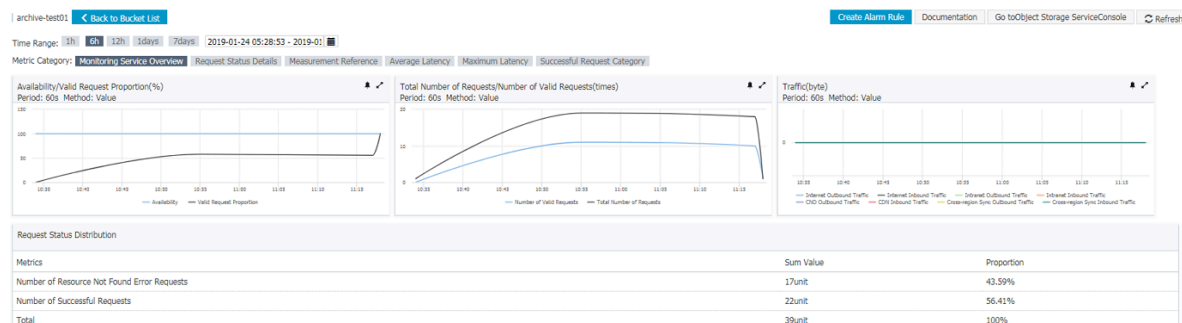
The Bucket list tab page displays the information including bucket name, region, creation time, metering statistics of the current month, and related operations.

								Monthly Data (Deadline: 2019.01.24 10:00:00)
Users		Bucket List		Alarm Rules				
	Name	Region	Created At	Storage Size	Internet Outbound Traffic	Number of Get requests	Number of Put requests	Actions
<input type="checkbox"/>	bucket-1	China North 2 (Beijing)	2018-04-30 13:14:04	774.17KB	0B	186times	30times	Monitoring Charts Alarm Rules
<input type="checkbox"/>	bucket-2	Asia Pacific SE 3 (Kuala Lumpur)	2018-02-19 16:25:28	253.83KB	0B	1245times	11times	Monitoring Charts Alarm Rules
<input type="checkbox"/>	bucket-3	China North 2 (Beijing)	2018-01-30 17:28:33	6.16KB	0B	52times	50times	Monitoring Charts Alarm Rules
<input type="checkbox"/>	bucket-4	China North 2 (Beijing)	2018-01-30 19:26:23	0B	0B	690times	20times	Monitoring Charts Alarm Rules
<input type="checkbox"/>	bucket-5	China East 1 (Hangzhou)	2017-08-18 14:09:18	42.84GB	4.74MB	211times	20times	Monitoring Charts Alarm Rules
<input type="checkbox"/>	bucket-6	China East 1 (Hangzhou)	2018-02-12 10:30:32	2.38MB	0B	0times	0times	Monitoring Charts Alarm Rules
<input type="checkbox"/>	bucket-7	China East 1 (Hangzhou)	2018-05-22 10:55:44	2.57GB	0B	0times	0times	Monitoring Charts Alarm Rules
<input type="checkbox"/>	bucket-8	China North 2 (Beijing)	2018-05-08 15:47:57	0B	0B	18times	0times	Monitoring Charts Alarm Rules
<input type="checkbox"/>	bucket-9	China North 2 (Beijing)	2017-11-27 17:36:12	361.00KB	0B	0times	0times	Monitoring Charts Alarm Rules
<input type="checkbox"/>	bucket-10	US East 1 (Virginia)	2018-05-08 15:49:24	0B	0B	0times	0times	Monitoring Charts Alarm Rules
Set Alarm Rules								Total 22yunjankong.common.pageInfo.unit 10 2 3

- Display parameters are as follows: The metering statistics of the current month display the storage size, Internet outbound traffic, Put request count, and Get request count for each bucket.
- Click Monitoring chart or the corresponding bucket name to go to the bucket monitoring view page.
- Click Alarm rules next to your expected bucket, or go to the Alarm rules tab to display all alarm rules of the bucket.
- Enter the expected bucket name in the search box in the upper left-corner to display the bucket (fuzzy match is supported).
- Select the check boxes before the expected bucket names and click Setting custom monitor alarm rules to batch set alarm rules. For more information, see the [Alarm Service User Guide](#).

- **Bucket-level monitoring view**

Click Monitoring chart next to the expected bucket name in the bucket list to go to the bucket monitoring view.



The bucket monitoring view displays metric charts based on the following six indicator groups:

- **Monitoring Service Overview**
- **Request Status Details**
- **Measurement Reference**
- **Average Latency**
- **Maximum Latency**
- **Successful Request Category**

Except measurement reference, other indicators are displayed with an aggregation granularity of 60s. The default time range for bucket-level metric charts is of the previous six hours, whereas that for user-level metric charts is of the previous hour. Click Back to bucket list in the upper-left corner to return to the Bucket list tab.

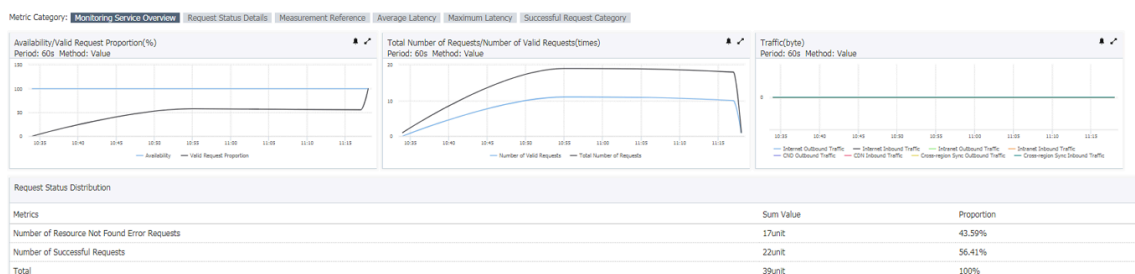
- **Monitoring Service Overview**

This indicator group is similar to the service monitoring overview at the user level, but the former displays metric data at the bucket level. The main metric charts include:

- **Request Valid Availability**, which includes two metric indicators: availability and percentage of valid requests
- **Total/Valid request**, which includes two metric indicators: total number of requests and number of valid requests
- **Overflow**, which includes eight metric indicators: Internet outbound traffic, Internet inbound traffic, intranet outbound traffic, intranet inbound traffic,

CDN outbound traffic, CDN inbound traffic, outbound traffic of cross-region replication, and inbound traffic of cross-region replication

- Request status count, which is a table that displays the number and percentage of each type of requests within the selected time range.



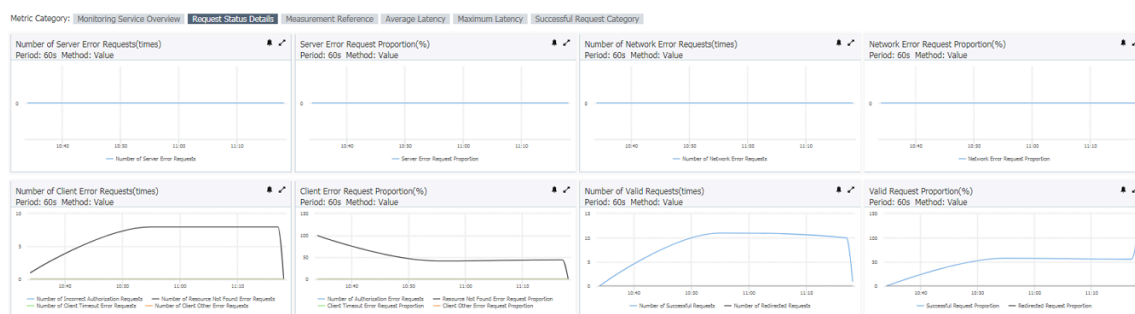
- Request Status Details

This indicator group is similar to the request state details at the user level, but the former displays metric data at the bucket level. The main metric charts include:

- Server error count
- Server error rate
- Network error count
- Network error request rate
- Client error request count, which includes four metric indicators: number of error requests indicating resource not found, number of authorization error requests, number of client-site time-out error requests, and number of other client-site error requests
- Client error request percent, which includes four metric indicators: percentage of error requests indicating resource not found, percentage of

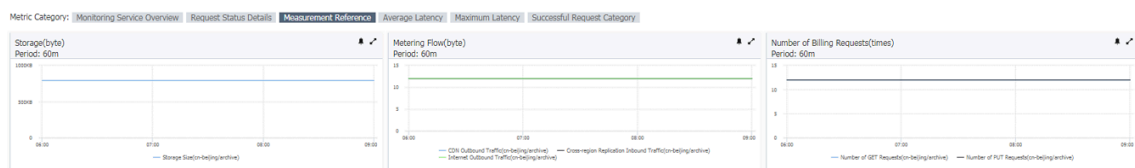
authorization error requests, percentage of client-site time-out error requests, and percentage of other client-site error requests

- Redirect request count, which includes two metric indicators: number of successful requests and number of redirect requests
- Success redirect rate, which includes two metric indicators: percentage of successful requests and percentage of redirect requests



- Measurement Reference

The metering reference group shows metering indicators with an hourly collection and representation granularity, as shown in the following figure:



The metering metric charts include:

- Quota size
- Overflow
- Billing requests, which includes the Get request count and Put request count.

After a bucket is created, new data is collected in the next hour on the hour following the current time point, and the collected data will be displayed within 30 minutes.

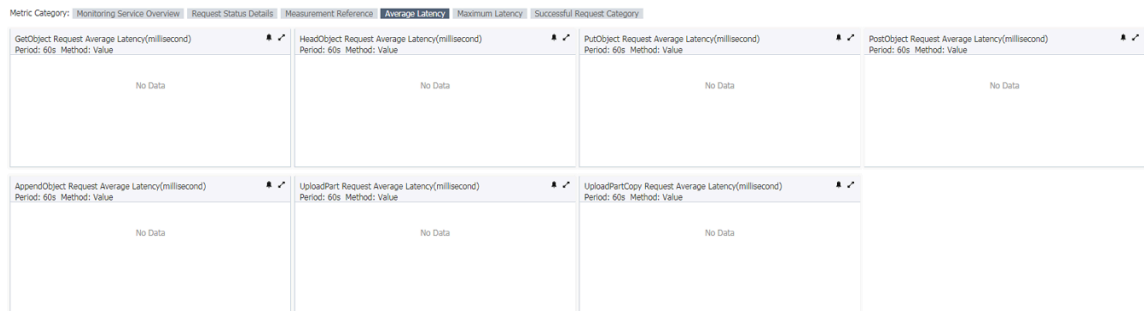
- Average Latency

This indicator group contains the average latency indicators of API monitoring. The metric charts include:

- getObject Average Latency
- headObject Average Latency
- putObject Average Latency

- **postObject Average Latency**
- **append Object Average Latency**
- **upload Part Average Latency**
- **upload Part Copy Average Latency**

Each metric chart shows the corresponding average E2E latency and average server latency. See the figure below:

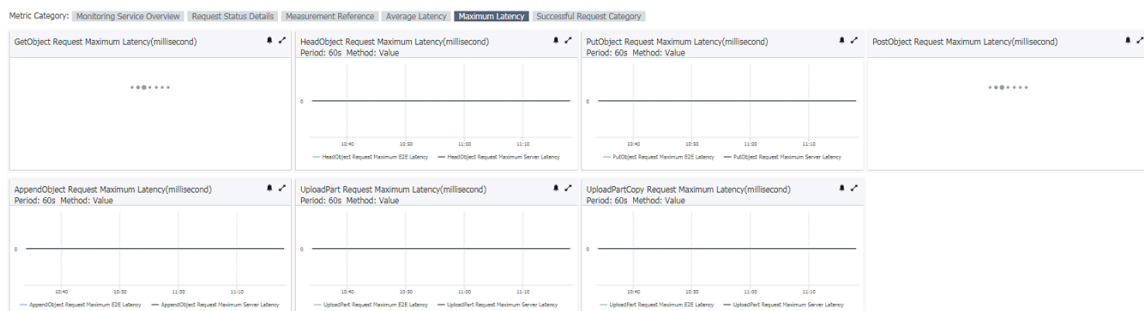


- **Maximum Latency**

This indicator group contains the maximum latency indicators of API monitoring. The metric charts include:

- **getObject Max Latency(Millisecond)**
- **headObject Max Latency**
- **putObject Max Latency**
- **postObject Max Latency**
- **append Object Max Latency**
- **upload Part Max Latency**
- **upload Part Copy Max Latency**

Each metric chart shows the corresponding maximum E2E latency and maximum server latency. See the following figure:

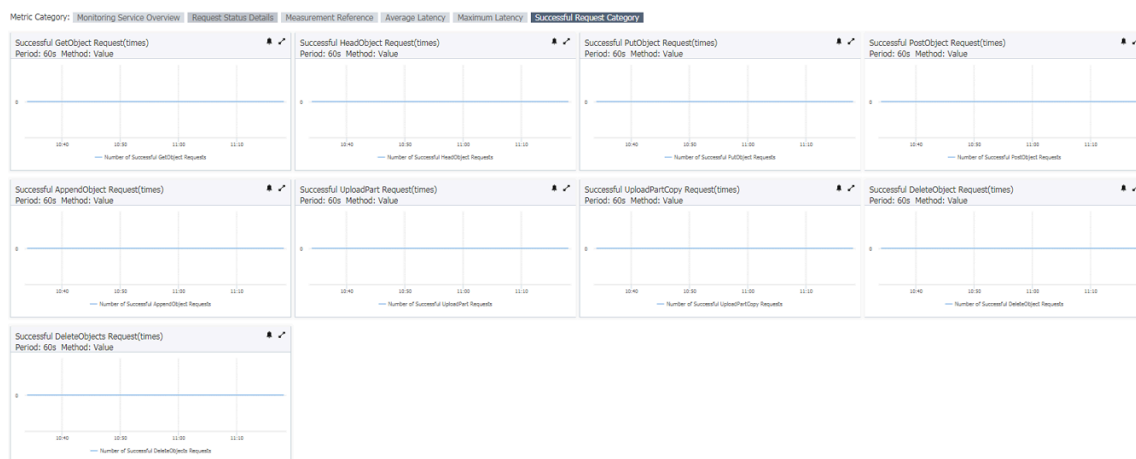


- **Successful Request Category**

This indicator group contains the successful request count indicators of API monitoring. The metric charts include:

- getObject Success Count
- headObject Success Count
- putObject Success Count
- post Object Success Count
- append Object Success Count
- upload Part Success Count
- upload Part Copy Success Count
- delete Object Success Count
- deleteObjects Success Count

See the following figure:



Alarm Rules

The Alarm rules tab page allows you to view and manage all your alarm rules, as shown in the following figure:

Users Bucket List Alarm Rules						
All	Please select.					
Rule Name	Status (All)	Enable	Metrics (All)	Dimensions (All)	Alarm Rules	Notification Contact
appendobject	OK	Enabled	Number of Successful AppendObject Requests	resource:_ALL	Number of Successful AppendObject Requests >=10000 Info Give an alarm 1 consecutive times	Default Co... View
						View Alarm Logs Modify Disable Delete
<input type="button" value="Enable"/> <input type="button" value="Disable"/> <input type="button" value="Delete"/>						Total 1yunjankong.common.pageInfo.unit 10 <input type="button" value="Previous"/> <input type="button" value="Next"/>

For the description and usage of the "Alarm Rules" tab page, see the [Alarm Service User Guide](#).

Additional links

For more information regarding the important points and user guide of the monitoring service, see the related chapter in [Monitoring, diagnosis, and troubleshooting](#).

19.3 Alert service

This topic describes alert rules for monitoring OSS in the CloudMonitor console and how to set such alert rules.

Before getting started with OSS alert rules, you can read the following topics to get familiar with the basic concepts of the alert service and the configurations of alert contacts and alert contact groups.

- [Alert service overview](#)
- [Manage alert contacts and alert contact groups](#)

OSS alert rules are developed based on OSS metrics. Therefore, OSS alert rules are categorized by dimensions similar to those of OSS metrics. Two alert dimensions are available: user-level and bucket-level.

Alarm Rules tab

The CloudMonitor console provides an [Alarm Rules](#) tab for OSS monitoring and alerting. On this tab, you can view, modify, enable, disable, and delete alert rules. You can also view the historical alert information of a specific alert rule.

User overview

Bucket list

Alarm rules

All

* Please select

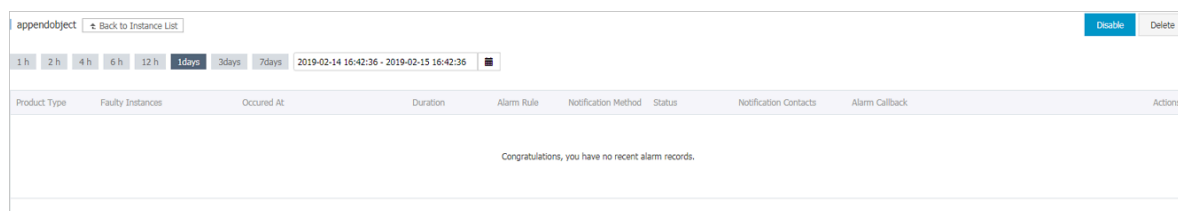
Alarm dimension	Monitored items	Sampling period	Alarm rules	Alarm type	Alarm	Enable St				
<input type="checkbox"/> User level	User internet send	5 minutes	Monitoring value it alarms1times continuously,> 9byte	aobeitest2... View	OK	Yes	Alarm history	Modify	Suspend	Delete
<input checked="" type="checkbox"/> User level	User success count	5 minutes	Monitoring value it alarms1times continuously,> 15yunjiankong.metric.unitName.frequency	aobeitest2... View	OK	Yes	Alarm history	Modify	Suspend	Delete
<input type="checkbox"/> User level	User total request count	5 minutes	Monitoring value it alarms1times continuously,> 10yunjiankong.metric.unitName.frequency	aobeitest2... View	OK	Yes	Alarm history	Modify	Suspend	Delete
<input type="checkbox"/> User level	User valid request count	5 minutes	Monitoring value it alarms1times continuously,> 10yunjiankong.metric.unitName.frequency	aobeitest2... View	OK	Yes	Alarm history	Modify	Suspend	Delete
<div><div><input type="checkbox"/> enable</div><div>Forbidden</div><div>Delete</div></div>										

Total:4, per page showing30

<1>

- Click View in the Actions column corresponding to an alert rule to view its content.
- Click Modify in the Actions column corresponding to an alert rule to modify it.
- Click Delete in the Actions column corresponding to an alert rule to delete it. You can also select multiple alert rules and click Delete below the alert rule list to delete multiple alert rules at a time.

- If an alert rule is in the `Enabled` state, click `Disable` in the `Actions` column corresponding to the alert rule to disable it. After the alert rule is disabled, you no longer receive alert information for this rule. You can also select multiple alert rules and click `Disable` below the alert rule list to disable multiple alert rules at a time.
- If an alert rule is in the `Disabled` state, click `Enable` in the `Actions` column corresponding to the alert rule to enable it. The alert rule takes effect again to detect exceptions and send alert information. You can also select multiple alert rules and click `Enable` below the alert rule list to enable multiple alert rules at a time.
- Click `Alarm Logs` in the `Actions` column corresponding to an alert rule. You can view the information about historical alerts corresponding to this rule.



Relevant concepts:

- Historical alert information records the past status changes of a selected alert rule, such as a status change from `OK` to `Alarm`, a status change from `Alarm` to `OK`, and a special status change of `Muted`.
- When an alert rule is in the `Muted` state, the alert triggered by this alert rule remains active within a specified period and is not cleared. During the muted period, the system does not send alert information to notification contacts until the muted period expires.
- Historical alert information is kept for one month. Alerts generated earlier than one month are automatically deleted. You can query data of three days at most, and cannot query data generated 31 days ago.

You can click `View` in the `Notification Contact` column corresponding to an alert rule to view the members of notification contacts (in alert contact groups) and the methods that they use to receive alert information (such as SMS message, email, or TradeManager), as shown in the following figure.

Alarm Contacts

Default Contact Group

Contact	Email ID	DingTalk Robot
Alib...	✓	

OK

View alert rules

- Locate alert rules

You can use the GUI elements on the Alarm Rules tab to locate the alert rules that you search for.

- Alert dimension drop-down list: You can select All or BucketLevel. If you select All, all user-level and bucket-level alert rules appear in the alert rule list.

Users Bucket List Alarm Rules

All * Please select.

- Bucket drop-down list: If you select BucketLevel from the alert dimension drop-down list, the bucket drop-down list displays all buckets of the current Alibaba Cloud account. You can select a bucket to view all alert rules for this bucket.

Users Bucket List Alarm Rules

BucketLevel * Please select.

- Metrics drop-down list: It displays all OSS metrics, including user-level and bucket-level metrics. If you select All, user-level or bucket-level alert rules for all metrics appear in the alert rule list.
- Status drop-down list: You can select a status to view all alert rules in this state, such as OK, Alarm, Insufficient Data, Enable, or Disable. If you select All, alert rules in all statuses appear in the alert rule list.

- View all alert rules

If you click the Alarm Rules tab, all alert rules automatically appear in the alert rule list. You can also select **All** from the alert dimension drop-down list to view all alert rules. Then, you can use the **Metrics** and **Status** drop-down lists to set filtering conditions and accurately locate alert rules in this alert dimension.

- View alert rules for a specific bucket

To view the alert rules for a specific bucket, you need to select **BucketLevel** from the alert dimension drop-down list and select the name of the destination bucket from the bucket drop-down list. You can also click the **Bucket List** tab. On the tab that appears, you can click **Alarm Rules** in the **Actions** column corresponding to the relevant bucket to go to the Alarm Rules tab, where you can view all alert rules for this bucket. Then, you can use the **Metrics** and **Status** drop-down lists to set filtering conditions and accurately locate alert rules in this alert dimension.

- View alert rules related to a specific metric



You can select a metric from the **Metrics** drop-down list to view all alert rules for this metric.

- View alert rules in a certain alert state

You can select an alert status from the **Status** drop-down list, such as **Alarm**, to view all alert rules that are in this status.

Add an alert rule

1. Use any of the following methods to go to the Create Alarm Rule page:

- On the [Users](#) tab, click **Monitoring Service Overview** and click  in any chart.
- On the [Bucket List](#) tab, click the relevant bucket name to go to the bucket details page. Click **Create Alarm Rule**.
- On the [Bucket List](#) tab, click the relevant bucket name. On the page that appears, click **Monitoring Service Overview** and click  in any chart.

2. Set the alert rule as needed.

- **Related Resource**
 - Products : **Select** Object Storage Service .
 - Resource Range : **Select** All Resources **or** bucketDimensions as needed.
 - Bucket (if you set Resource Range to bucketDimensions): **Select** one or more buckets as needed.
- **Set Alarm Rules**
 - **Alarm Rule:** Enter the alert rule name.
 - **Rule Describe:** Select the content, time, and threshold of the metric as needed.
 - **+Add Alarm Rule:** Click it to add more alert rules.
 - **Mute for:** Specify the interval for sending an alert notification if the exception persists after the alert is triggered.
 - **Triggered when threshold is exceeded for:** Specify the times that the rule is matched consecutively to send alert notifications. For example, if you select Internet Outbound Traffic, 1mins, Value, >, and 100 for Rule Describe and set Triggered when threshold is exceeded for to 3, an alert is triggered only when the Internet outbound traffic exceeds 100 MB three consecutive times within 1 minute.
 - **Effective Period :** Select the time the alert rule takes effect.
- **Notification Method**
 - **Notification Contact:** If you have set an alert contact group by following the procedure in [Manage alert contacts and alert contact groups](#), select the group. If you have not set any alert contact groups, click Quickly create a contact group to create a group by following the instructions.
 - **Notification Methods :** Select notification methods for the alert rule.
 - **Email Subject:** Enter the subject of the notification email.
 - **Email Remark:** optional. Enter the remarks of the email.
 - **HTTP CallBack:** Enter a URL that can be accessed from the Internet. CloudMonitor sends a POST request to push the alert notification to this URL. Currently, only HTTP is supported.

3. Click Confirm to complete the alert rule setting.

Notes

Currently, alert rules for a bucket are not associated with the existence of the bucket. If you delete a bucket, alert rules for this bucket still exist. Therefore, we recommend that you delete alert rules for a bucket before deleting this bucket.

19.4 Metric item reference

This chapter provides parameter references to use with the API, or the CloudMonitor SDK, to access the metric data of the OSS monitoring service.

Space

The OSS monitoring service metric data uses the same Namespace: `acs_oss`.

Sample code written by the Java SDK:

```
QueryMetricRequest request = new QueryMetricRequest ();  
request.setNamespace ("acs_oss");
```

StartTime and EndTime

The value range of the time parameters for CloudMonitor is in the format of [StartTime, EndTime]. The data that is attributed to StartTime is not collected, whereas the data that is attributed to EndTime can be accessed.

The CloudMonitor retention policy specifies that data is retained for 31 days. This means the interval between StartTime and EndTime cannot exceed 31 days, and data outside the 31 day collection period cannot be accessed.

For more information about other time parameters, see [CloudMonitor API Reference](#).

Sample code written by the Java SDK:

```
request.setStartTime ("2016-05-15 08:00:00");  
request.setEndTime ("2015-05-15 09:00:00");
```

Dimensions

OSS metric items are classified into user level bucket level based on application scenarios. The value of Dimensions varies with regards to access of metric data at these different levels.

- Dimensions does not need to be set for access to user-level metric data.

- Set Dimensions access to bucket-level metric data as follows:

```
{" BucketName ": " your_bucket_name "}
```

`your_bucket_name` indicates the name of the bucket you want to access.

Note: Dimensions is a JSON string and has only one Key-Value pair for OSS metric indicators.

Sample code written by the Java SDK:

```
request . setDimensions ("{" BucketName ":" your_bucket_name "}" );
```

Period

The aggregation granularity of all OSS metric indicators, except metering indicators, is 60s by default. The aggregation granularity of metering indicators is 3,600s by default.

Sample code written by the Java SDK:

```
request . setPeriod (" 60 " );
```

Metric

The [Monitoring indicators reference](#) describes the following metric items.

Metric	Metric item name	Unit	Level
Useravailability	User-level availability	%	User level
UserRequestValidRate	User-level valid request rate	%	User level
UserTotalRequestCount	User-level requests	Times	User level
UserValidRequestCount	User-level valid requests	Times	User level
UserInternetSend	User-level Internet outbound traffic	Byte	User level
UserInternetRecv	User-level Internet inbound traffic	Byte	User level
UserIntranetSend	User-level intranet outbound traffic	Byte	User level

Metric	Metric item name	Unit	Level
UserIntranetRecv	User-level intranet inbound traffic	Byte	User level
UserCdnSend	User-level CDN outbound traffic	Byte	User level
UserCdnRecv	User-level CDN inbound traffic	Byte	User level
UserSyncSend	User-level outbound traffic of cross-region replication	Byte	User level
UserSyncRecv	User-level inbound traffic of cross-region replication	Byte	User level
UserServerErrorCount	User-level server-site error requests	Times	User level
UserServerErrorRate	User-level server-site error request rate	%	User level
UserNetworkErrorCount	User-level network-site error requests	Times	User level
UserNetworkErrorRate	User-level network-site error request rate	%	User level
UserAuthorizationErrorCount	User-level client-site authorization error requests	Times	User level
UserAuthorizationErrorRate	User-level client-site authorization error request rate	%	User level
UserResourceNotFoundErrorCount	User-level client-site error requests indicating resource not found	Times	User level
UserResourceNotFoundErrorRate	User-level client-site error request rate indicating resource not found	%	User level

Metric	Metric item name	Unit	Level
UserClientTimeoutErrorCount	User-level client-site time-out error request	Times	User level
UserClientOtherErrorRate	User-level client-site time-out error request rate	%	User level
UserClientOtherErrorCount	Other user-level client-site error requests	Times	User level
UserClientOtherErrorRate	Other user-level client-site error request rate	%	User level
UserSuccessCount	Successful user-level requests	Times	User level
UserSuccessRate	Successful user-level request rate	%	User level
UserRedirectCount	User-level redirect requests	Times	User level
UserRedirectRate	User-level redirect request rate	%	User level
Availability	Availability	%	Bucket level
RequestValidRate	Valid request rate	%	Bucket level
TotalRequestCount	Requests	Times	Bucket level
ValidRequestCount	Valid requests	Times	Bucket level
InternetSend	Internet outbound traffic	Byte	Bucket level
InternetRecv	Internet inbound traffic	Byte	Bucket level
IntranetSend	Intranet outbound traffic	Byte	Bucket level
IntranetRecv	Intranet inbound traffic	Byte	Bucket level
CdnSend	CDN outbound traffic	Byte	Bucket level
CdnRecv	CDN inbound traffic	Byte	Bucket level

Metric	Metric item name	Unit	Level
SyncSend	Outbound traffic of cross-region replication	Byte	Bucket level
SyncRecv	Inbound traffic of cross-region replication	Byte	Bucket level
ServerErrorCount	Server-site error requests	Times	Bucket level
ServerErrorRate	Server-site error request rate	%	Bucket level
NetworkErrorCount	Network-site error requests	Times	Bucket level
NetworkErrorRate	Network-site error request rate	%	Bucket level
AuthorizationErrorCount	Client-site authorization error requests	Times	Bucket level
AuthorizationErrorRate	Client-site authorization error request rate	%	Bucket level
ResourceNotFoundErrorCount	Client-site error requests indicating resource not found	Times	Bucket level
ResourceNotFoundErrorRate	Client-site error request rate indicating resource not found	%	Bucket level
ClientTimeoutErrorCount	Client-site time-out error requests	Times	Bucket level
ClientTimeoutErrorRate	Client-site time-out error request rate	%	Bucket level
ClientOtherErrorCount	Other client-site error requests	Times	Bucket level
ClientOtherErrorRate	Other client-site error request rate	%	Bucket level
SuccessCount	Successful requests	Times	Bucket level

Metric	Metric item name	Unit	Level
SuccessRate	Successful request rate	%	Bucket level
RedirectCount	Redirect requests	Times	Bucket level
RedirectRate	Redirect request rate	%	Bucket level
GetObjectE2eLatency	Average E2E latency of GetObject requests	Millisecond	Bucket level
GetObjectServerLatency	Average server latency of GetObject requests	Millisecond	Bucket level
MaxGetObjectE2eLatency	Maximum E2E latency of GetObject requests	Millisecond	Bucket level
MaxGetObjectServerLatency	Maximum server latency of GetObject requests	Millisecond	Bucket level
HeadObjectE2eLatency	Average E2E latency of HeadObject requests	Millisecond	Bucket level
HeadObjectServerLatency	Average server latency of HeadObject requests	Millisecond	Bucket level
MaxHeadObjectE2eLatency	Maximum E2E latency of HeadObject requests	Millisecond	Bucket level
MaxHeadObjectServerLatency	Maximum server latency of HeadObject requests	Millisecond	Bucket level
PutObjectE2eLatency	Average E2E latency of PutObject requests	Millisecond	Bucket level

Metric	Metric item name	Unit	Level
PutObjectServerLatency	Average server latency of PutObject requests	Millisecond	Bucket level
MaxPutObjectE2ELatency	Maximum E2E latency of PutObject requests	Millisecond	Bucket level
MaxPutObjectServerLatency	Maximum server latency of PutObject requests	Millisecond	Bucket level
PostObjectE2ELatency	Average E2E latency of PostObject requests	Millisecond	Bucket level
PostObjectServerLatency	Average server latency of PostObject requests	Millisecond	Bucket level
MaxPostObjectE2ELatency	Maximum E2E latency of PostObject requests	Millisecond	Bucket level
MaxPostObjectServerLatency	Maximum server latency of PostObject requests	Millisecond	Bucket level
AppendObjectE2ELatency	Average E2E latency of AppendObject requests	Millisecond	Bucket level
AppendObjectServerLatency	Average server latency of AppendObject requests	Millisecond	Bucket level
MaxAppendObjectE2ELatency	Maximum E2E latency of AppendObject requests	Millisecond	Bucket level
MaxAppendObjectServerLatency	Maximum server latency of AppendObject requests	Millisecond	Bucket level

Metric	Metric item name	Unit	Level
UploadPartE2eLatency	Average E2E latency of UploadPart requests	Millisecond	Bucket level
UploadPartServerLatency	Average server latency of UploadPart requests	Millisecond	Bucket level
MaxUploadPartE2eLatency	Maximum E2E latency of UploadPart requests	Millisecond	Bucket level
MaxUploadPartServerLatency	Maximum server latency of UploadPart requests	Millisecond	Bucket level
UploadPartCopyE2eLatency	Average E2E latency of UploadPartCopy requests	Millisecond	Bucket level
UploadPartCopyServerLatency	Average server latency of UploadPartCopy requests	Millisecond	Bucket level
MaxUploadPartCopyE2eLatency	Maximum E2E latency of UploadPartCopy requests	Millisecond	Bucket level
MaxUploadPartCopyServerLatency	Maximum server latency of UploadPartCopy requests	Millisecond	Bucket level
GetObjectCount	Successful GetObject requests	Times	Bucket level
HeadObjectCount	Successful HeadObject requests	Times	Bucket level
PutObjectCount	Successful PutObject requests	Times	Bucket level

Metric	Metric item name	Unit	Level
PostObjectCount	Successful PostObject requests	Times	Bucket level
AppendObjectCount	Successful AppendObject requests	Times	Bucket level
UploadPartCount	Successful UploadPart requests	Times	Bucket level
UploadPartCopyCount	Successful UploadPartCopy requests	Times	Bucket level
DeleteObjectCount	Successful DeleteObject requests	Times	Bucket level
DeleteObjectsCount	Successful DeleteObjects requests	Times	Bucket level

The following table lists the metric items of metering indicators with an aggregation granularity of 3,600s.

Metric	Metric item name	Unit	Level
MeteringStorageUtilization	Size of storage	Byte	If Dimensions is set, the returned metric data belongs to the bucket level; if Dimensions is not set, the returned metric data belongs to the user level.
MeteringGetRequest	Get requests	Times	If Dimensions is set, the returned metric data belongs to the bucket level; if Dimensions is not set, the returned metric data belongs to the user level.

Metric	Metric item name	Unit	Level
MeteringPutRequest	Put requests	Times	If Dimensions is set, the returned metric data belongs to the bucket level; if Dimensions is not set, the returned metric data belongs to the user level.
MeteringInternetTx	Volume of Internet outbound traffic	Byte	If Dimensions is set, the returned metric data belongs to the bucket level; if Dimensions is not set, the returned metric data belongs to the user level.
MeteringCdnTX	Volume of CDN outbound traffic	Byte	If Dimensions is set, the returned metric data belongs to the bucket level; if Dimensions is not set, the returned metric data belongs to the user level.
MeteringSyncRX	Volume of inbound traffic of cross-region replication	Byte	If Dimensions is set, the returned metric data belongs to the bucket level; if Dimensions is not set, the returned metric data belongs to the user level.

Sample code written by the Java SDK:

```
request.setMetric("UserAvailability");
```

19.5 Monitoring indicators reference

OSS indicators can be monitored at the user level or the bucket level based on application scenarios.

In addition to common chronological metric indicators, the system analyzes and collects statistics on the existing metric indicators for easy user observation of metric data and matching of billing policy. Statistical indicators over a specified period of time are provided, such as request status distribution and metering statistics of the month. This reference guide describes the indicators in detail.

All indicators are collected at the minute-level (per minute) except for metering and statistical indicators. Metering indicators are collected at the hour-level (per hour).

User-level indicators

The user level indicator refers to the indicator information that monitors the overall situation of the OSS system used from the user's account level, and is a summary of all bucket related monitoring data under the account. User-level indicators consist of three monitoring indicator details: current-month metering statistics, service monitoring overview, and request state details.

Service monitoring overview

Indicators in service monitoring overview are basic service indicators. Details of service monitoring overview indicators are as follows:

Indicator	Unit	Description
Availability	%	An indicator showing the system availability of using the storage service. It is obtained through the equation: $1 - \frac{\text{percentage of requests with server - end errors (indicated by a return code 5xx)}}{\text{in all requests}}$.
Valid requests rate	%	Percentage of valid requests in all requests. For more information about valid requests, see the following description.
Requests	Times	Total number of requests received and processed by the OSS server

Indicator	Unit	Description
Valid requests	Times	Total number of requests whose return code is 2xx or 3xx.
Internet outbound traffic	Byte	Downstream Internet traffic
Internet inbound traffic	Byte	Upstream Internet traffic
Intranet outbound traffic	Byte	Downstream intranet traffic of the service system
Intranet inbound traffic	Byte	Upstream intranet traffic of the service system
CDN outbound traffic	Byte	Downstream CDN traffic when CDN acceleration service is activated, that is , the origin retrieval traffic
CDN inbound traffic	Byte	Upstream CDN traffic when CDN acceleration service is activated
Outbound traffic of cross-region replication	Byte	Downstream traffic generated in the data replication process when the cross-region replication function is activated
Inbound traffic of cross-region replication	Byte	Upstream traffic generated in the data replication process when the cross-region replication function is activated

In addition to the above specific monitoring indicators, statistics are also provided for the distribution of request status over a period of time. The statistics are mainly based on the status codes of the returned status codes or OSS error codes (the total number and the percentage of requests within the observed time period).

Request state details

Request state details indicators are requested monitoring information based on the return status code, or OSS error code, associated with the different requests. Details of request state details indicators are as follows:

Indicator	Unit	Description
Server-site error requests	Times	Total number of requests with system-level errors indicated by a return code 5xx
Server-site error requests rate	%	Percentage of requests with server-end errors in all requests
Network error requests	Times	Total number of requests whose <i>HTTP status code is 499</i>
Network error requests rate	%	Percentage of requests with network errors in all requests
Client-end authorization error requests	Times	Total number of requests with a return code 403
Client-end authorization error requests rate	%	Percentage of requests with client-end authorization errors in all requests
Client-end error requests indicating resource not found	Times	Total number of requests with a return code 404
Client-end error requests rate indicating resource not found	%	Percentage of requests with client-end errors indicating resource not found in all requests
Client-end time-out error requests	Times	Total number of requests whose return status code is 408 or return OSS error code is RequestTimeout
Client-end time-out error requests rate	%	Percentage of requests with client-end time-out errors in all requests
Other client-end error requests	Times	Total number of requests with other client-end errors indicated by a return code 4xx

Indicator	Unit	Description
Other client-end error requests rate	%	Percentage of requests with other client-end errors in all requests
Successful requests	Times	Total number of requests whose return code is 2xx.
Successful requests rate	%	Percentage of successful requests in all requests
Redirect requests	Times	Total number of requests whose return code is 3xx.
Redirect requests rate	%	Percentage of redirect requests in all requests

Current-month metering statistics

Metering statistics of the current month are collected from 00:00 on the first day of the month to the metering cutoff time as indicated in the same month.

Details of the metering indicators currently available are as follows:

Indicator	Unit	Description
Storage size	Byte	Size of the total storage occupied by all buckets of a specified user before the metering statistic collection deadline.
Internet outbound traffic	Byte	Total Internet outbound traffic of the user from 00:00 of the first day of the current month to the metering statistic collection deadline.
Put requests	Times	Total number of Put requests of the user from 00:00 of the first day of the current month to the metering statistic collection deadline.

Indicator	Unit	Description
Get requests	Times	Total number of Get requests of the user from 00:00 of the first day of the current month to the metering statistic collection deadline.

Bucket-level indicators

Bucket-level indicators are used to monitor OSS operations of specific buckets and are applicable for business scenarios. In addition to current-month metering statistics and basic service indicator items such as service monitoring overview and request state details (which can be monitored at the account level), bucket-level indicators include metering indicators and performance indicators such as metering reference, latency, and successful request operation categories.

Service monitoring overview

Similar to the user-level description, the service monitoring overview indicators are basic indicators, but use metric data that is displayed at the bucket-level.

Request state details

Similar to the user-level description, the request state details indicators use metric data that is displayed at the bucket-level.

Current-month metering statistics

Statistical methods are similar to those listed in current-month metering statistics at the user level, but the former collects resource usage statistics at the bucket level.

Details of current-month metering statistics at the bucket-level are as follows:

Indicator	Unit	Description
Storage size	Byte	Size of storage occupied by a specified bucket before the metering statistic collection deadline.

Indicator	Unit	Description
Internet outbound traffic	Byte	Total Internet outbound traffic of a specified bucket from 00:00 of the first day of the current month to the metering statistic collection deadline.
Put requests	Times	Total number of Put requests of a specified bucket from 00:00 of the first day of the current month to the metering statistic collection deadline.
Get requests	Times	Total number of Get requests of a specified bucket from 00:00 of the first day of the current month to the metering statistic collection deadline.

Metering indicators

Metering indicators are monitored chronologically, and are collected and aggregated at the hour-level. Details of metering indicators are as follows:

Indicator	Unit	Description
Storage size	Byte	Average size of storage used by a specified bucket in an hour.
Internet outbound traffic	Byte	Total Internet outbound traffic of a specified bucket in an hour.
Put requests	Times	Total number of Put requests of a specified bucket in an hour.
Get requests	Times	Total number of Gut requests of a specified bucket in an hour.

Latency

Latency Request latency directly reflects the system performance and is monitored using two indicators: average latency and maximum latency. The indicators are collected and aggregated at the minute-level. Moreover, indicators can be classified based on the OSS API request operation type to more specifically reflect the performance of the system responding to different operations. Only APIs involving data operations in bucket-related operations (excluding meta operations) are monitored currently.

Besides, in order to facilitate analyzing performance hotspots and environmental problems, latency monitoring indicators are collected from two different links of E2E and the server, in which:

- E2E latency refers to the E2E latency of sending a successful request to OSS, including the processing time OSS requires to read the request, send a response, and receive a response confirmation message.
- Server latency is the latency of OSS processing a successful request, excluding the network delay involved in E2E latency.

Note that performance indicators are used to monitor successful requests (with a return status code 2xx).

The following table lists specific metric indicator items:

Indicator	Unit	Description
Average E2E latency of GetObject requests	Millisecond	Average E2E latency of successful requests whose request API is GetObject
Average server latency of GetObject requests	Millisecond	Average server latency of successful requests whose request API is GetObject
Maximum E2E latency of GetObject requests	Millisecond	Maximum E2E latency of successful requests whose request API is GetObject
Maximum server latency of GetObject requests	Millisecond	Maximum server latency of successful requests whose request API is GetObject

Indicator	Unit	Description
Average E2E latency of HeadObject requests	Millisecond	Average E2E latency of successful requests whose request API is HeadObject
Average server latency of HeadObject requests	Millisecond	Average server latency of successful requests whose request API is HeadObject
Maximum E2E latency of HeadObject requests	Millisecond	Maximum E2E latency of successful requests whose request API is HeadObject
Maximum server latency of HeadObject requests	Millisecond	Maximum server latency of successful requests whose request API is HeadObject
Average E2E latency of PutObject requests	Millisecond	Average E2E latency of successful requests whose request API is PutObject
Average server latency of PutObject requests	Millisecond	Average server latency of successful requests whose request API is PutObject
Maximum E2E latency of PutObject requests	Millisecond	Maximum E2E latency of successful requests whose request API is PutObject
Maximum server latency of PutObject requests	Millisecond	Maximum server latency of successful requests whose request API is PutObject
Average E2E latency of PostObject requests	Millisecond	Average E2E latency of successful requests whose request API is PostObject
Average server latency of PostObject requests	Millisecond	Average server latency of successful requests whose request API is PostObject
Maximum E2E latency of PostObject requests	Millisecond	Maximum E2E latency of successful requests whose request API is PostObject

Indicator	Unit	Description
Maximum server latency of PostObject requests	Millisecond	Maximum server latency of successful requests whose request API is PostObject
Average E2E latency of AppendObject requests	Millisecond	Average E2E latency of successful requests whose request API is AppendObject
Average server latency of AppendObject requests	Millisecond	Average server latency of successful requests whose request API is AppendObject
Maximum E2E latency of AppendObject requests	Millisecond	Maximum E2E latency of successful requests whose request API is AppendObject
Maximum server latency of AppendObject requests	Millisecond	Maximum server latency of successful requests whose request API is AppendObject
Average E2E latency of UploadPart requests	Millisecond	Average E2E latency of successful requests whose request API is UploadPart
Average server latency of UploadPart requests	Millisecond	Average server latency of successful requests whose request API is UploadPart
Maximum E2E latency of UploadPart requests	Millisecond	Maximum E2E latency of successful requests whose request API is UploadPart
Maximum server latency of UploadPart requests	Millisecond	Maximum server latency of successful requests whose request API is UploadPart
Average E2E latency of UploadPartCopy requests	Millisecond	Average E2E latency of successful requests whose request API is UploadPartCopy

Indicator	Unit	Description
Average server latency of UploadPartCopy requests	Millisecond	Average server latency of successful requests whose request API is UploadPart Copy
Maximum E2E latency of UploadPartCopy requests	Millisecond	Maximum E2E latency of successful requests whose request API is UploadPart Copy
Maximum server latency of UploadPartCopy requests	Millisecond	Maximum server latency of successful requests whose request API is UploadPartCopy

Successful request operation categories

In conjunction with latency monitoring, the monitoring of successful requests reflects the system capability of processing access requests to a certain extent. Similarly, only APIs involving data operations in bucket-related operations are monitored currently. The following lists specific indicator items:

Indicator	Unit	Description
Successful GetObject requests	Times	Number of successful requests whose request API is GetObject
Successful HeadObject requests	Times	Number of successful requests whose request API is HeadObject
Successful PutObject requests	Times	Number of successful requests whose request API is PutObject
Successful PostObject requests	Times	Number of successful requests whose request API is PostObject
Successful AppendObject requests	Times	Number of successful requests whose request API is AppendObject
Successful UploadPart requests	Times	Number of successful requests whose request API is UploadPart

Indicator	Unit	Description
Successful UploadPart Copy requests	Times	Number of successful requests whose request API is UploadPartCopy
Successful DeleteObject requests	Times	Number of successful requests whose request API is DeleteObject
Successful DeleteObjects requests	Times	Number of successful requests whose request API is DeleteObjects

19.6 Service monitoring, diagnosis, and troubleshooting

Despite reducing users' costs of infrastructure construction and O&M cloud applications compared to traditional applications, cloud applications have complicated monitoring, diagnosis, and troubleshooting. The OSS storage service provides a wide array of monitoring and log information, helping you fully understand program behavior and promptly discover and locate problems.

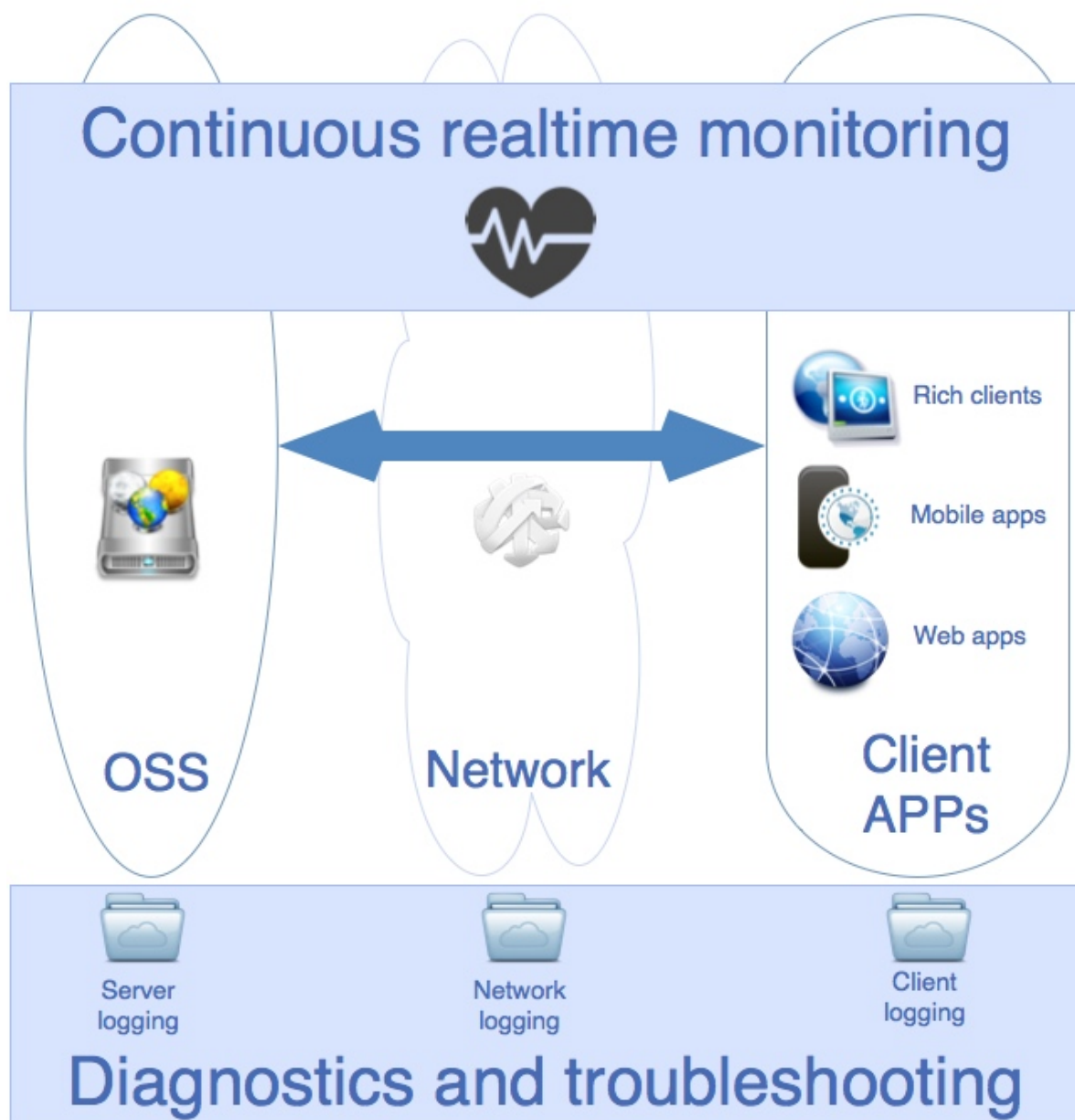
Overview

This chapter instructs you how to monitor, diagnose, and troubleshoot OSS problems by using the OSS monitoring service, logging, and other third-party tools, helping you achieve the following goals:

- Monitors in real time the running status and performance of OSS and provides prompt alarm notifications.
- Provides effective methods and tools to help you locate problems.
- Provides methods to help you quickly solve common OSS-related problems.

This chapter is organized as follows:

- [OSS real-time monitoring](#): Describes how to use the OSS monitoring service to continuously monitor the running status and performance of OSS.
- [Tracking and diagnosis](#): Describes how to use the OSS monitoring service and logging function to diagnose problems, and how to associate the relevant information in log files for tracking and diagnosis.
- [Troubleshooting](#): Describes typical problems and corresponding troubleshooting methods.

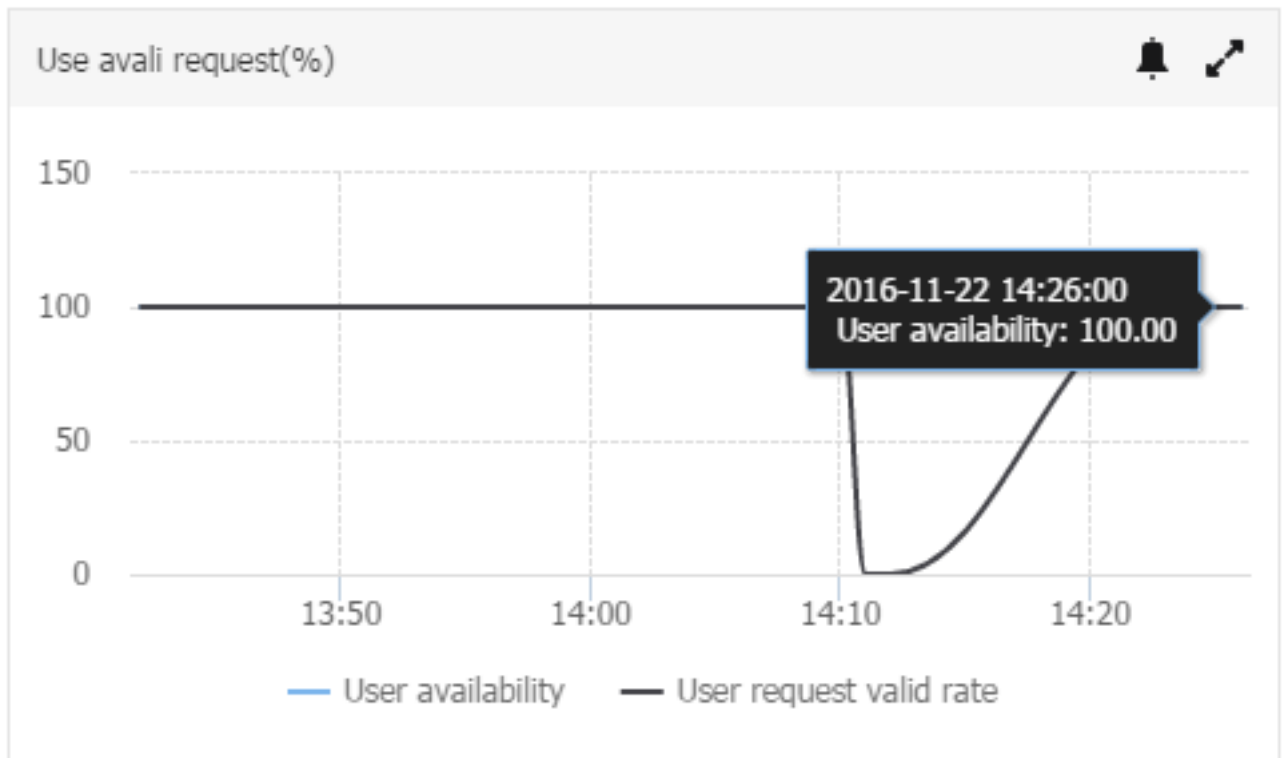


OSS monitoring

Overall operating conditions

- Availability and percentage of valid requests

This is an important indicator related to system stability and the ability of users to correctly use the system. Any value lower than 100% indicates that some requests have failed.



Availability may also temporarily fall below 100% due to system optimization factors, such as partition migration for load balancing. In these cases, OSS SDKs can provide relevant retry mechanisms to handle this type of intermittent failure, keeping the service end unaware.

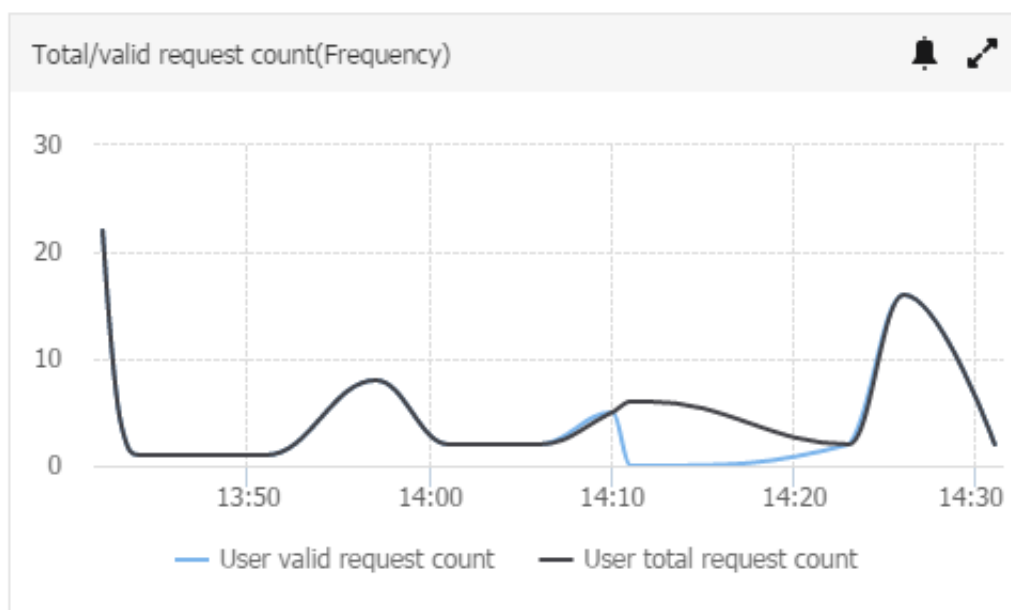
Also, when the percentage of valid requests falls below 100%, you must analyze the issue based on your own usage. You can use request distribution statistics or request status details to determine the actual types of request errors. Then, you can use [Tracking and Diagnosis](#) to determine the cause and perform [Troubleshooting](#). In some business scenarios, a valid request rate is expected to fall below 100%. For example, you may need to first check that an object exists and then perform a certain operation based on the existence of the object. In this case, if the object does not exist, the read request that checks its existence returns a 404 error code

(resource does not exist error). This inevitably produces a valid request rate of less than 100%.

For businesses that require high system availability, you can set an alarm rule that is triggered when the indicator falls below the expected threshold value.

- Total No. of requests and No. of valid requests

This indicator reflects the system operation status from the perspective of the total traffic volume. When the No. of valid requests is not equal to the total No. of requests, this indicates that some requests have failed.



You can watch the fluctuations in the total No. of requests and No. of valid requests, especially when they sharply increase or decrease. In such cases, follow-up action is required. You can set alarm rules to make sure you receive prompt notifications. For periodic businesses, you can set periodic alarm rules (periodic alarms will be available soon). For more information, see [Alarm Service User Guide](#).

- Request status distribution statistics

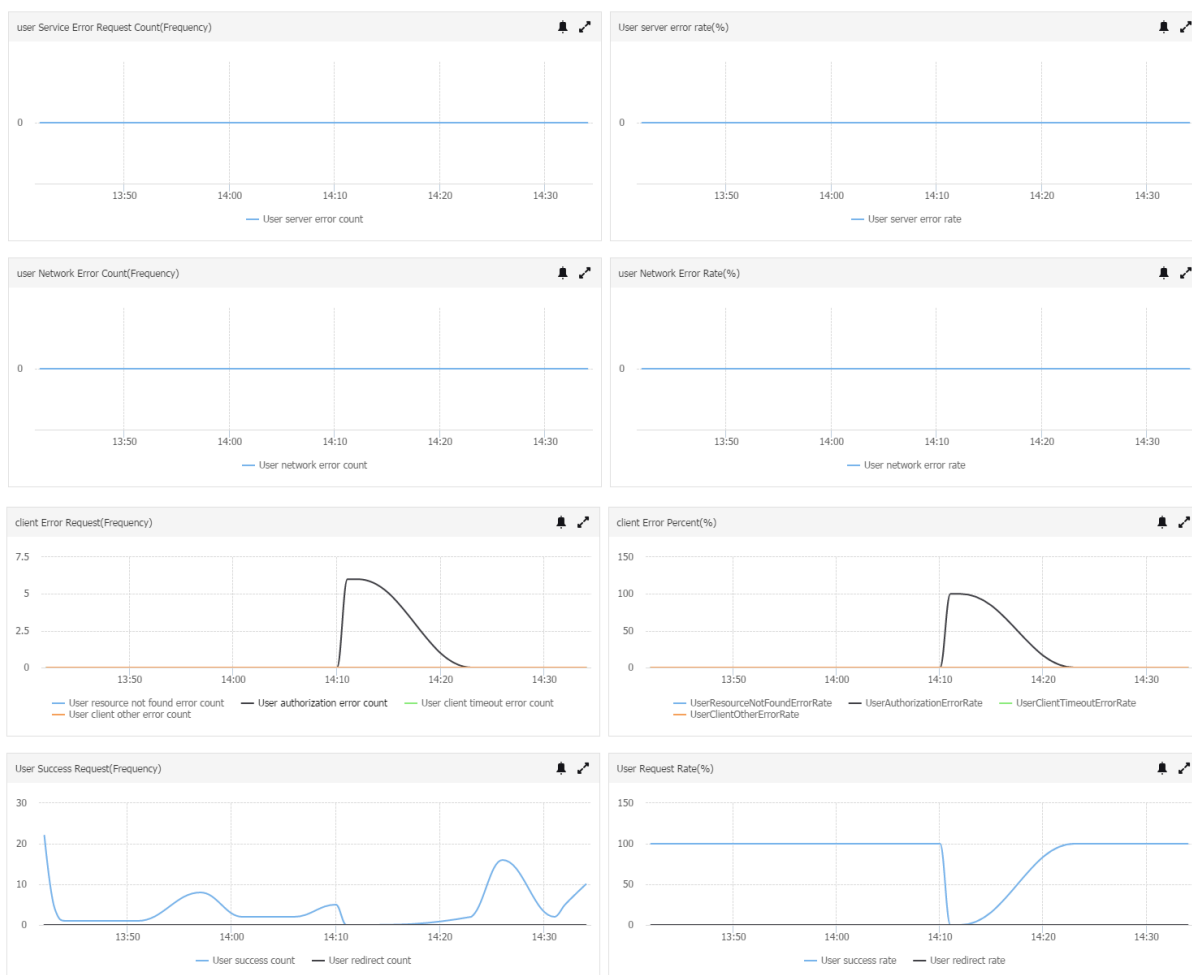
When availability or the valid request rate falls below 100% (or the No. of valid requests is not equal to the total No. of requests), you can look at the request status

distribution statistics to quickly determine the request error types. For more information about this metric indicator, see [OSS Metric Indicator Reference Manual](#).

User level request		
Metric	Sum value	Percent
User authorization error count	12yunjiankong.metric.unitName.frequency	14.29%
User success count	72yunjiankong.metric.unitName.frequency	85.71%
Sum	84yunjiankong.metric.unitName.frequency	100%

Request status details monitoring

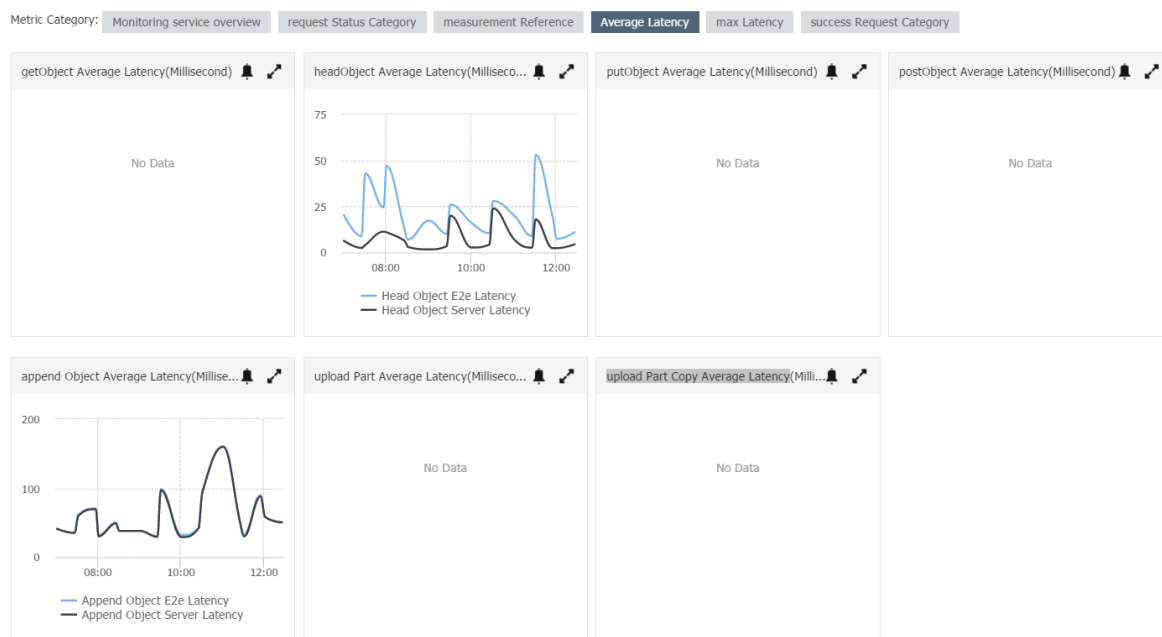
Request status details provides more details about the request monitoring status on the basis of request status distribution statistics. They let you monitor certain types of requests in more detail.



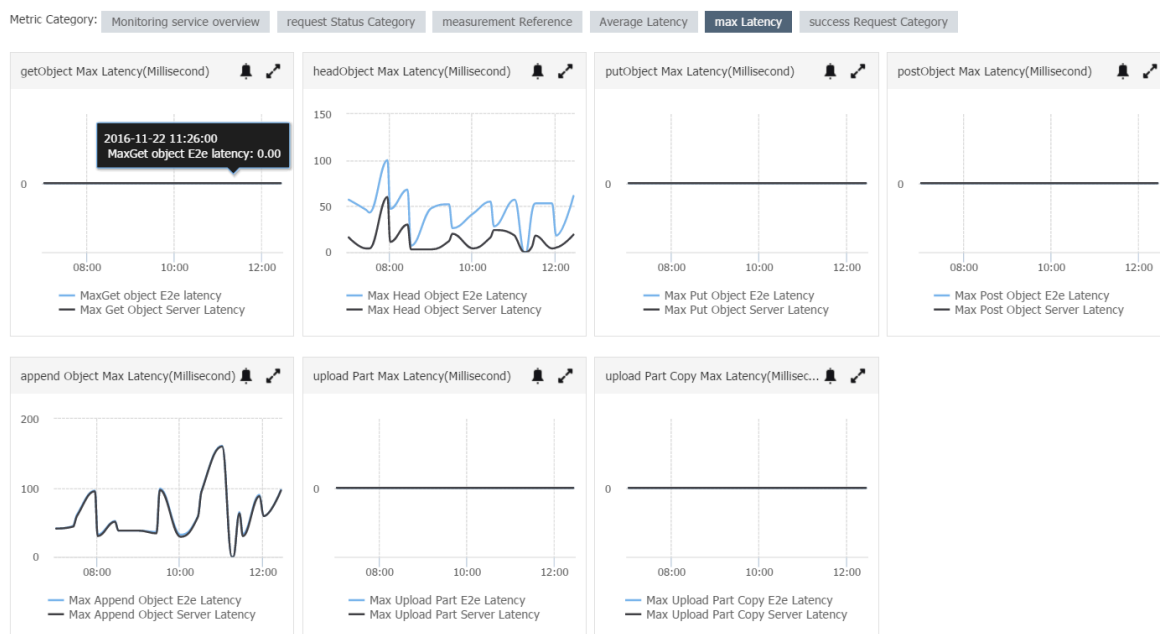
Performance monitoring

The monitoring service provides the following metric items that can be used as indicators for performance monitoring.

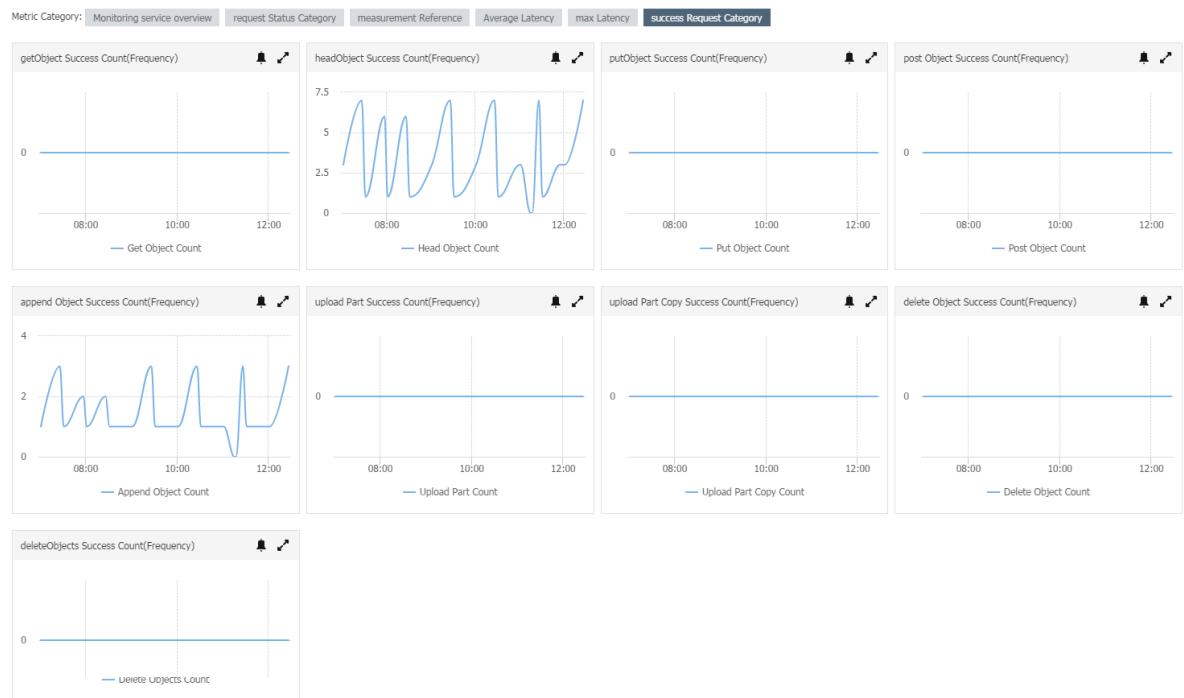
• **Average latency: E2E average latency and Server average latency**



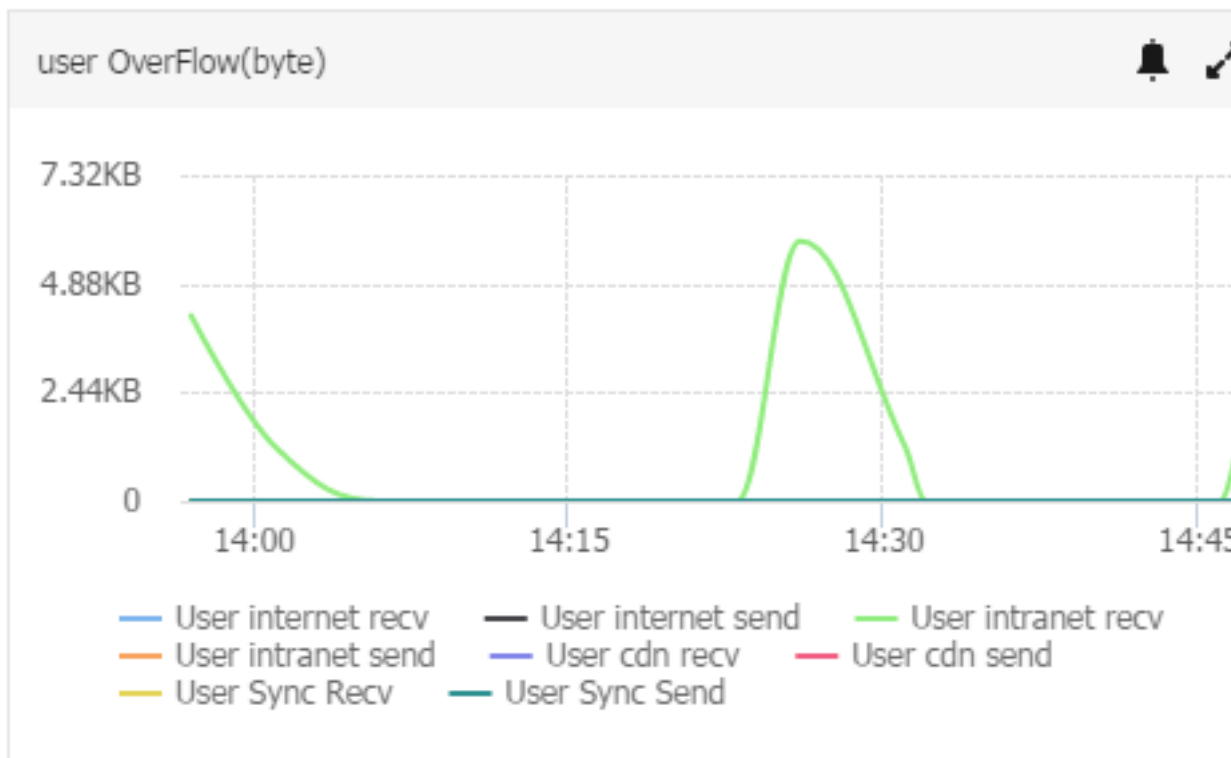
• **Maximum latency: E2E maximum latency and Server maximum latency**



· Successful request categories



· Traffic



The preceding metric items (except for ‘Traffic’) implement categorized monitoring based on API operation types:

- GetObject
- HeadObject
- PutObject
- PostObject
- AppendObject
- UploadPart
- UploadPartCopy

The latency indicators show the average or maximum time needed for API operation types to process requests. E2E latency is the indicator for end-to-end latency. Besides the time needed to process requests, it also includes the time needed to read requests and send responses, and the delay caused by network transmission. Server latency only includes the time needed to process the requests on the server, not the client-side transmission network latency. Therefore, if the E2E latency suddenly increases but the server latency does not change significan

tly, you can determine that the poor performance has been caused by network instability, instead of an OSS system fault.

In addition to the APIs mentioned previously, "successful request operation categories" also monitors the quantity of requests for the following two API operation types:

- DeleteObject
- Deleteobjects

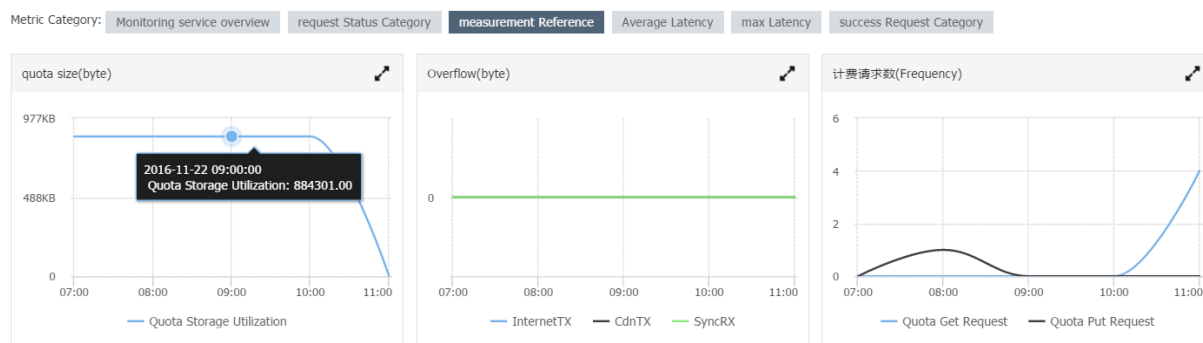
The traffic indicator is used to monitor the overall situation for a user or a specific bucket. It looks at the usage of network resources in Internet, intranet, CDN origin retrieval, cross-domain replication, and other such scenarios.

For performance-type indicators, we must focus on sudden and abnormal changes, such as when the average latency suddenly spikes or remains above the normal request latency baseline for a long period of time. You can set alarm rules that correspond to performance indicators, so that the relevant personnel are immediately notified if an indicator falls below or exceeds a threshold value. For businesses with periodic peaks and troughs, you can set periodic alarm rules for week on week, day on day, or hour on hour comparisons (periodic alarms will be available soon).

Billing monitoring

At press time, the OSS monitoring service can only monitor storage space, outbound Internet traffic, Put requests, and Get requests (not including cross-domain replication outbound traffic and CDN outbound traffic). It does not support alarm setting or API read operations for billing data.

The OSS monitoring service collects bucket-level billing monitoring data on an hourly basis. In the monitoring view for a specific bucket, you can see graphs of continuous monitoring trends. Using the monitoring view, you can analyze your businesses' OSS resource usage trends and estimate future costs. See the following figure:



The OSS monitoring service also provides statistics on the quantity of user and bucket-level resources consumed each month. For example, the total amount of OSS resources consumed by an account or bucket starting from the 1st day of the month. These statistics are updated hourly. This increases your understanding of your resource usage and computation fees for the current month in real time, as shown in the following figure:

as shown in the following figure:



Note:

In the monitoring service, the provided billing data is pushed to the maximum extent possible, but this may cause some discrepancies with the actual bill amount. Please note that the Billing Center data is used in actual billing applications.

Tracking and diagnosis

Problem diagnosis

· Performance diagnosis

Many subjective factors are involved in the determination of application performance. You must use the satisfaction of your business needs in your specific business scenario as a baseline, to determine if a performance problem occurs. Also, when a client initiates a request, factors that may cause performance problems may come from anywhere in the request chain. For example, problems may be caused by OSS overloads, client TCP configuration problems, or traffic bottlenecks in the basic network architecture.

Therefore, when diagnosing performance problems, you must first set a reasonable baseline. Then, you use the performance indicators provided by the monitoring service to determine the potential root cause of any performance

problem. Next, you find detailed information in the relevant logs to help you further diagnose and troubleshoot any faults.

In the [Troubleshooting](#) section, we give examples of many common performance problems and troubleshooting measures. This can be used as a reference.

- Error diagnosis

When requests from client applications are at fault, the clients receive error information from the server. The monitoring service records these errors and shows statistics for the various types of errors that may affect requests. You can also retrieve detailed information for individual requests from the server log, client log, and network log. Generally, the returned HTTP status code, OSS error code, and OSS error information can indicate the cause of the request failure.

For error response information details, see [OSS error responses](#).

- Using the logging function

OSS provides a server logging function for user requests. This helps you track end-to-end detailed request logs.

For instructions on the activation and use of the logging function, see [Set logging](#).

For more information on Log Service naming rules and record formats, see [Set access logging](#).

- Using network logging tools

In many situations, you can diagnose problems by using the logging function to record storage log and client application log data. However, in certain situations, you may need more details by using network logging tools.

This is because capturing traffic exchanged between clients and the server can give you more detailed information on the data exchanged between clients and server and the underlying network conditions, which can help you investigate problems. For example, in some situations, user requests may report an error, but no request can be seen in the server log. In such cases, you can use the records logged by the OSS logging function to see if the cause of the problem lies with the client, or you can use network monitoring tools to check for a network problem.

[Wireshark](#) is one of the most common network log analysis tools. This free protocol analyzer runs on the packet level and provides a view of detailed packet

information for various network protocols. This can help you troubleshoot packet loss and connection problems.

see [Wireshark User Guide](#).

E2E tracking and diagnosis

Requests are initiated by a client application process and pass through the network environment to the OSS server, where they are processed. Then, a response is sent by the server over the network environment and received by the client. This is an end-to-end tracking process. Associating client application logs, network tracking logs, and server logs provides detailed information for you to troubleshoot the root cause of a problem and discover potential problems.

In OSS, the provided RequestIDs serve as identifiers used to associate the information from various logs. In addition, the log timestamps not only allow you to quickly query specific log time ranges, but can also show you the time points when request events and other client application, network, and service system events occurred during this period. This helps you analyze and investigate problems.

· RequestID

Whenever the OSS receives a request, it allocates it a unique server request ID, its RequestID. In different logs, the RequestID is located in different fields:

- In server logs recorded by the OSS logging function, the RequestID is located in the “Request ID” column.
- In the process of network tracking (for example, when using Wireshark to capture data streams), the RequestID is the x-oss-request-id header value in the response message.
- In client applications, you must use the client code to manually print the RequestID in the client log. At the press time, the latest Java SDK version already supported printing RequestID information for normal requests. You can use the `getRequestId` operation to retrieve RequestIDs from the results returned by different APIs. All OSS SDK versions allow you to print RequestIDs for abnormal requests. You can call the `OSSEException`'s `getRequestId` method to obtain this information.

- **Timestamps**

You can use timestamps to find relevant log entries. You must note that there may be some deviations between the client time and server time. On a client, you can use timestamps to search for server log entries recorded by the logging function. For this, you must add or subtract 15 minutes.

Troubleshooting

Common performance-related problems

- **High average E2E latency, with low average server latency**

We have already discussed the differences between average E2E latency and average server latency. Therefore, we can say that high E2E latency and low server latency are caused by two possible reasons:

- Slow client application response speed
- Network factors

A slow client application response speed can be caused by several possible reasons:

- Limited number of available connections or threads
 - Use the relevant command to check if the system has a large number of connections in the TIME_WAIT status. If yes, adjust the core parameters to solve this problem.
 - When the number of available threads is limited, first check for bottlenecks affecting the client CPU, memory, network, or other resources. If no bottleneck is found, increase the number of concurrent threads properly.
 - If the problem persists, you have to optimize the client code. For example, you can use an asynchronous access method. You can also use the performance analysis function to analyze client application hotspots, and then perform the necessary optimization.
- Insufficient resources, such as CPU, memory, or bandwidth
 - For this type of problem, you must first use the relevant system monitoring function to find client resource bottlenecks. Then, optimize the client code

to rationalize resource usage or increase the client resources (increase the number of cores or the memory).

Investigate network latency problems

Generally, high E2E latency due to network factors is temporary. You can use Wireshark to investigate temporary and persistent network problems, such as packet loss problems.

- Low average E2E latency, low average server latency, but high client request latency

When the client experiences high request latency, the most probable cause is that the requests are not reaching the server. Therefore, we must find out why the client requests are not arriving at the server.

Two client-side factors can cause high client request sending latency:

- A limited number of available connections or threads: see the solution described in the preceding section.
- Client requests are retried multiple times: In this situation, you must find and solve the cause of the request retries based on the retry information. You can follow these steps to determine if the client has a retry problem:
 - Check the client log. The detailed log entries indicate if retries have occurred. Using the OSS Java SDK as an example, you can search for the following warn or info-level log entries. If such entries are found in the log, this indicates that requests have been retried.

```
[ Server ] Unable to execute HTTP request :  
Or  
[ Client ] Unable to execute HTTP request :
```

- If the client log level is debug, search for the following log entries (again we are using the OSS Java SDK as an example). If such entries exist, this indicates requests have been retried.

```
Retrying on
```

If no problem with the client occurs, you must check for potential network problems, such as packet loss. You can use a tool such as Wireshark to investigate network problems.

- High average server latency

If the server latency during downloads or uploads is high, this may be caused by the following two factors:

- A large number of clients are frequently accessing the same small object.

In this situation, you can view the server log recorded by the logging function to determine if a small object or a group of small objects are being frequently accessed in a short period of time.

For download scenarios, we suggest you activate the CDN service for this bucket , to improve performance. This also reduces your traffic fees. In the case of upload, you may consider revoking write permissions for this object (bucket), if this does not affect your business.

- Internal system factors

For internal system problems or problems that cannot be solved through optimization, please provide our system staff with the RequestIDs in your client logs or in the logs recorded by the logging function, and they can help you solve the problem.

Server errors

When the number of server-side errors increases, two scenarios must to be considered:

- Temporary increase

For this type of problem, you must adjust the retry policy in the client program and adopt a reasonable concession mechanism, such as exponential backoff. This not only avoids temporary service unavailability due to system optimization, upgrades , and other such operations (such as partition migration for system load balancing), but also avoids high pressure during business peaks.

- Permanent increase

When the number of server-side errors sustainably increases, please provide our back-end staff with the RequestIDs in your client logs or in the logs recorded by the logging function, and they can help you find the problem.

Network errors

Network errors occur when the server is processing a request and the connection is lost (not due to a server-side issue), so the HTTP request header cannot be returned.

In such a situation, the system records an *HTTP Status Code of 499* for this request. In the following situations, the server may change the request status code to 499:

- Before processing a received read/write request, if the server detects that the connection is unavailable, the request is recorded as 499.
- When the server is processing a request and the client preemptively closes the connection, the request is recorded as 499.

In summary, a network error occurs during the request process when a client independently closes the request or the client is disconnected from the network. If the client independently closes requests, you can check the client code, to identify the cause and time of the client's disconnection from OSS. When the client loses its network connection, you can use a tool such as Wireshark to investigate network connection problems.

Client errors

- Increase in client authorization errors

If you detect an increase in client authorization errors or the client receives a large number of 403 request errors, this is most commonly caused by the following problems:

- The bucket domain name accessed by the user is incorrect.
 - If the user uses a third-level or second-level domain name to access a bucket, this may cause a 403 error if the bucket is not in the region indicated by the domain name. For example, if you have created a bucket in the Hangzhou region, but a user attempts to access it using the domain name `Bucket.oss-cn-shanghai.aliyuncs.com`. In this case, you must confirm the bucket's region and then correct the domain name information.
 - If you have activated the CDN acceleration service, this problem may occur when CDN binds an incorrect origin retrieval domain name. In this case, check that the CDN origin retrieval domain name is the bucket's third-level domain name.
- If you encounter 403 errors when using JavaScript clients, this may be caused by a problem in the CORS (Cross-Origin Resource Sharing) settings, because web browsers implement "same source policy" security restrictions. In this

case, you must check the bucket's CORS settings and correct any errors. For information about CORS settings, see [CORS](#).

- Access control problems can be divided into four types:
 - When you use a primary AK for access, you must check the AK settings for errors if the AK is invalid.
 - When you use a RAM sub-account for access, you must check that the sub-account is using the correct sub-account AK and that the sub-account has the relevant permissions.
 - When you use temporary STS tokens for access, you must confirm that the temporary token has not expired. If the token has expired, apply for a new one.
 - If you use bucket or object settings for access control, you must check that the bucket or object to be accessed supports the relevant operations.
- When you authorize third-party downloads (using signed URLs to access OSS resources), if access was previously normal and then suddenly reports a 403 error, it is likely that the URL has expired.
- When RAM sub-accounts use OSS utilities, this may also produce 403 errors. These utilities include ossftp, ossbrowser, and the OSS console client. When you enter the relevant AK information during logon and the system throws an error, if you entered the correct AK, you must check that the AK is a sub-account AK and that this sub-account has permission for GetService and other operations.
- Increase in client-side 'resource does not exist' errors

When the client receives a 404 error, this means that you are attempting to access a resource or information that does not exist. When the monitoring service detects an increase in 'resource does not exist' errors, this is most likely caused by one of the following problems:

- Service usage: For example, when you first need to check that an object exists before performing another operation and you call the `doesObjectExist` method (using the Java SDK as an example), if the object does not exist, the client receives

the value "false". However, the server actually produces a 404 request error. Therefore, in this business scenario, 404 errors are normal.

- The client or another process previously deleted this object. You can confirm this problem by searching for the relevant object operation in the server log recorded by the logging function.
- Network faults case packet loss and retries. For example, the client may initiate a delete operation to delete a certain object. The request reaches the server and successfully executes the delete operation. However, if the response packet is lost during transmission on the network, the client initiates a retry. This second request then produces a 404 error. You can confirm that network problems are producing 404 errors using the client log and server log:
 - Check for retry requests in the client application log.
 - Check if the server log shows two delete operations for this object and that the first delete operation has an HTTP status of 2xx.
- Low valid request rate and high number of other client-side request errors

The valid request rate is the number of requests that return an HTTP status code of 2xx/3xx as a percentage of total requests. Status codes of 4XX or 5XX indicate a failed request and reduce the valid request rate. Other client-side request errors indicate requests errors other than the following: server errors (5xx), network errors (499), client authorization errors (403), resource does not exist errors (404), and client time-out errors (408 or OSS error code: RequestTimeout 400).

Check the server log recorded by the logging function to determine the specific errors encountered by these requests. You can see [OSS error responses](#) to find a list of common error codes returned by OSS. Then, check the client code to find and solve the specific cause of these errors.

Abnormal increase in storage capacity

If storage capacity increases abnormally without a corresponding increase in upload requests, this is generally caused by a delete problem. In such a case, check for the following two factors:

- When the client application uses a specific process to regularly delete storage objects to free up space: The investigation processes for this request are as follows:
 1. Check if the valid request rate has decreased, because a failed delete request may cause storage objects to fail to be deleted as expected.
 2. Find the specific cause for the decrease in the valid request rate by looking at the error types of the requests. Then, you can combine the specific client logs to see the detailed error information (for example, the STS temporary token used to free up storage space may have expired).
- When the client sets a LifeCycle to delete storage objects: Use the console or an API to check that the current bucket LifeCycle value is the same as before. If not, modify the configuration and use the server log recorded by the logging function to find information on the previous modification of this value. If the LifeCycle is normal but inactive, contact an OSS system administrator to help identify the problem.

Other OSS problems

If the Troubleshooting section did not cover your problem, use one of the following methods to diagnose and troubleshoot the problem.

1. View the OSS monitoring service, to see if there have been any changes compared to the expected baseline behavior. Using the monitoring view, you may be able to determine if this problem is temporary or permanent and which storage operations are affected.
2. The monitoring information can help you search the server log data recorded by the logging function, to find information on any errors that may have occurred when the problem started. This information may be able to help you find and solve the problem.
3. If the information in the server log is insufficient, use the client log to investigate the client application, or use a network tool such as Wireshark to check your network for problems.

20 Cloud data processing

Image Processing

For introduction and more information about functions, see [Image Processing](#).

Media Processing

Media Processing is a transcoding computing service for multimedia data. It provides an economic, easy-to-use, elastic, and highly scalable method for conversion of audio and video stored on OSS into formats suitable for playing on PCs, TVs, or mobile devices.

Media Processing was constructed based on Alibaba Cloud computing services. In the past, users had to make a high investment to purchase, build, and manage transcoding software and hardware, and perform complex configuration optimization, transcoding parameter adaptation, and other operations. Media Processing has transformed everything. It has enhanced the elasticity of cloud computing services. Media Processing offers transcoding capabilities to fulfill business transcoding demands to its extreme and also curbs the wastage of resources.

Media Processing functions include the Web management console, service APIs, and SDKs. Users can use and manage Media Processing and integrate transcoding functions into their own apps and services.

Media Processing function list

- Transcoding
- Pipelines
- Screenshot
- Media information
- Watermark
- Preset templates
- Custom templates
- Video clip output
- Resolution scaling
- M3U8 custom segment length output
- Audio/Video extraction

- Video image rotation
- Video-to-GIF conversion

For introduction and more information about functions, see [Media Processing documentation](#).

21 Hide

21.1 Access control

21.1.1 Bucket permission separation

Another scenario is introduced in this section. If another user is using the developed app, you can use an individual bucket to store your app data. Assume that the bucket is the ram-test-app. In consideration of permission separation, the application server must not be allowed to access the ram-test-app; that is, the account ram_test_pub is permitted only to read ram-test-dev. This can also be realized through the RAM permission system. The procedure is as follows:

1. Because the system has no default bucket-level policy, we must create a custom policy.

The bucket access policy is shown as follows. For more information, see [RAM Policy Description](#) and [OSS Authorization FAQ](#).

```
{
  "Version": " 1 ",
  "Statement": [
    {
      "Effect": " Allow ",
      "Action": [
        " oss : ListObject s ",
        " oss : GetObject "
      ],
      "Resource": [
        " acs : oss :*:ram - test - dev ",
        " acs : oss :*:ram - test - dev /*"
      ]
    }
  ]
}
```

After setting, we can see the policy in the custom authorization policy list.

2. In user authorization management, add this policy to the selected authorization policy list. Also in Users > Management > Authorization policy, all previously granted OSS read permissions can be revoked.

3. Test the validity of permission configured.

- The object in ram-test-dev can be accessed:

```

$./ osscmd get oss :// ram - test - dev / test . txt test .
txt -- host = oss - cn - hangzhou . aliyuncs . com - i o0hue
***** Frogv - k OmVwFJ03qc T0 ***** Fh0Ypg3p0K nA
100 % The object test . txt is downloaded to test .
txt , please check .
0 . 047 ( s ) elapsed

```

- The object in ram-test-app cannot be accessed:

```

$./ osscmd get oss :// ram - test - app / test . txt test .
txt -- host = oss - cn - hangzhou . aliyuncs . com - i o0hue
***** Frogv - k OmVwFJ03qc T0 ***** Fh0Ypg3p0K nA
Error Headers :
[(' content - length ', ' 229 '), (' server ', ' AliyunOSS '), ('
connection ', ' keep - alive '), (' x - oss - request - id ', '
5646ED53F9 EEA2F33241 91A2 '), (' date ', ' Sat , 14 Nov
2015 08 : 14 : 11 GMT '), (' content - type ', ' applicatio
n / xml ')]
Error Body :
<? xml version =" 1 . 0 " encoding =" UTF - 8 "? >
< Error >
  < Code > AccessDeni ed </ Code >
  < Message > AccessDeni ed </ Message >
  < RequestId > 5646ED53F9 EEA2F33241 91A2 </ RequestId >
  < HostId > ram - test - app . oss - cn - hangzhou . aliyuncs .
com </ HostId >
</ Error >
Error Status :
403
get Failed !

```

- Files cannot be uploaded to oss-test-app:

```

$./ osscmd put test . txt oss :// ram - test - app / test .
txt -- host = oss - cn - hangzhou . aliyuncs . com - i o0hue
***** Frogv - k OmVwFJ03qc T0 ***** Fh0Ypg3p0K nA

100 % Error Headers :
[(' content - length ', ' 229 '), (' server ', ' AliyunOSS '), ('
connection ', ' keep - alive '), (' x - oss - request - id ', '
5646ED7BB8 DE437A912D C7A8 '), (' date ', ' Sat , 14 Nov
2015 08 : 14 : 51 GMT '), (' content - type ', ' applicatio
n / xml ')]
Error Body :
<? XML version = " 1 . 0 " encoding = " UTF - 8 "? >
< Error >
  < Code > AccessDeni ed </ Code >
  < Message > AccessDeni ed </ Message >
  < RequestId > 5646ED7BB8 DE437A912D C7A8 </ RequestId >
  < HostId > ram - test - app . oss - cn - hangzhou . aliyuncs .
com </ HostId >
</ Error >
Error status :
403

```

```
put    Failed !
```

Using the preceding configuration, we have successfully separated the permissions for ram-test-dev and ram-test-app.

The preceding section explains how to use the subaccount permission control function to separate permissions and minimize the potential risk of information leakage.

If you want to implement more complex access control, see [RAM User Guide](#).

21.1.2 STS temporary access authorization

In the previous documents, we used only the RAM user functions. These user accounts are for long-term normal use. This poses as a serious risk if the RAM user permissions cannot be promptly revoked in case of information leakage.

In the previous example, assume that our developer's app allows users to upload data to the OSS bucket am-test-app and currently, the number of app users is large. In this case, how can the app securely grant data upload permissions to many users and how can it be certain of storage isolation among multiple users?

In such scenarios, we need to grant users temporary access using STS. STS can be used to specify a complex policy that restricts specified users by only granting them the minimum necessary permissions.

Create a role

Based on the example in the previous document, the app user has a bucket, ram-test-app, to store personal data. A role can be created as follows:

1. Create a RAM user account named ram_test_app using the process illustrated in the previous documents. Do not grant this account any permissions, because it inherits the permissions of a role which it assumes.
2. Create roles. Here you must create two roles for users to perform read operations and to upload files respectively.
 - Log on to the RAM console and select Roles > New Role.
 - Select a role type. Here you must select User role.
 - Enter the role type information. Because this role has been used by its own Alibaba Cloud account. Use the default setting.
 - Configure basic role information.

3. When the role was created, it did not have any permissions. Therefore, we must create a custom authorization policy using the process described earlier. The following is the authorization policy:

```
{
  "Version ": " 1 ",
  "Statement ": [
    {
      "Effect ": " allow ",
      "Action ": [
        " oss : ListObject s ",
        " Oss : GetObject "
      ],
      "Resource ": [
        " acs : oss :*:ram - test - app ",
        " acs : oss :*:ram - test - app /*"
      ]
    }
  ]
}
```

This indicates read-only permission for ram-test-app.

4. After the policy is established, give the role RamTestAppReadOnly the ram-test-app read-only permission on the role management page.
5. Perform the same procedure to create the role RamTestAppWrite and use a custom authorization policy to grant ram-test-app write permission. The authorization policy is as follows:

```
{
  "Version ": " 1 ",
  "Statement ": [
    {
      "Effect ": " Allow ",
      "Action ": [
        " oss : DeleteObje ct ",
        " oss : ListParts ",
        " oss : AbortMulti partUpload ",
        " oss : PutObject "
      ],
      "Resource ": [
        " acs : oss :*:ram - test - app ",
        " acs : oss :*:ram - test - app /*"
      ]
    }
  ]
}
```



```
}
```

Now we have created two roles, `RamTestAppReadOnly` and `RamTestAppWrite`, with read-only and write permissions for `ram-test-app`, respectively.

Temporary access authorization

After creating roles, we can use them to grant temporary access to OSS.

Preparation

Authorization is required for assuming roles. Otherwise, any RAM user could assume these roles, which can lead to unpredictable risks. Therefore, to assume corresponding roles, a RAM user needs to have explicitly configured permissions.

1. Create two custom authorization policies in authorization policy management.

Create Authorization Policy

STEP 1: Select an authorization policy

STEP 2: Edit permissions and

* Authorization policy name :

AliyunSTSAssumeRoleAccess201511160

The name must be 1-128 characters long numbers, and "-"

Remarks :

Policy content :

```

1 {
2   "Statement": [
3     {
4       "Action": "sts:AssumeRol
5       "Effect": "Allow",
6       "Resource":
7         "acs:ram::189[REDACTED]22283:r
8     }
9   ],
10  "Version": "1"

```

[Authorization policy format definition](#)
[Authorization policy FAQs](#)

```

{
  "Statement ": [
    {
      " Action ": " sts : AssumeRole ",
      " Effect ": " Allow ",

```

```

    " Resource ": " acs : ram :: 1894xxxxxx 722283 : role /
    ramtestapp readonly "
  },
  " Version ": " 1 "
}

```

Create another custom authorization policy using the same method:

```

{
  " Statement ": [
    {
      " Action ": " sts : AssumeRole ",
      " Effect ": " Allow ",
      " Resource ": " acs : ram :: 1894xxxxxx 722283 : role /
      ramtestapp write "
    }
  ],
  " Version ": " 1 "
}

```

Here, the content entered after Resource is a role' s ID. Role IDs can be found in Roles > Role Details .

2. Grant the two authorization policies to the account ram_test_app.

Use STS to grant access permissions

Now, we are ready with the platform to officially use STS to grant access permissions.

Here we use a simple STS Python command line tool [sts.py](#). The calling method is as follows:

```

$ python ./ sts . py AssumeRole RoleArn = acs : ram ::
1894xxxxxx 722283 : role / ramtestapp readonly RoleSessio nName
= usr001 Policy ='{" Version ":" 1 "," Statement ":[{" Effect ":"
Allow "," Action ":" oss : ListObject s "," oss : GetObject ","
Resource ":" acs : oss :*: ram - test - app "," acs : oss :*:
ram - test - app /*"}]}]' DurationSe conds = 1000 -- id = id --
secret = secret

```

- **RoleArn:** indicates the ID of a role to be assumed. Role IDs can be found in Roles > Role details .
- **RoleSessionName:** indicates the name of the temporary credentials. Generally, we recommend that you separate this using different application users.
- **Policy:** indicates a permission restriction, which is added when the role is assumed .
- **DurationSeconds:** indicate the validity time of the temporary credentials in seconds. The minimum value is 900, and the maximum value is 3600.
- **id and secret:** indicate the AccessKey of the RAM user to assume a role.

Here, we need to explain what is meant by “Policy” . The policy mentioned here is used to restrict the temporary credential permissions after a role is assumed . Ultimately, the permissions obtained by means of temporary credentials are overlapping permissions of the role and the policy passed in.

When a role is assumed, a policy can be entered to increase the flexibility. For example, when uploading the files, we can add different upload path restrictions for different users. This is shown in the following example.

Now, let's test the STS function. To test the bucket, first use the console to put the file test.txt in ram-test-app, with the content ststest.

Firstly, use the RAM user account ram_test_app to directly access the file. Next, replace AccessKey with your own access key used in the test.

```
[ admin @ NGIS - CWWF344M01 C / home / admin / oss_test ]
$./ osscmd get oss :// ram - test - app / test . txt test . txt
-- host = oss - cn - hangzhou . aliyuncs . com - i o0hue *****
Frogv - k OmVwFJ03qc T0 ***** Fh0Ypg3p0K nA
Error Headers :
[(' content - length ', ' 229 '), (' server ', ' AliyunOSS '), ('
connection ', ' keep - alive '), (' x - oss - request - id ', '
564A94D444 F4D8B2225E 4AFE '), (' date ', ' Tue , 17 Nov 2015
02 : 45 : 40 GMT '), (' content - type ', ' applicatio n / xml
')]
Error Body :
<? xml version =" 1 . 0 " encoding =" UTF - 8 "? >
< Error >
  < Code > AccessDeni ed </ Code >
  < Message > AccessDeni ed </ Message >
  < RequestId > 564A94D444 F4D8B2225E 4AFE </ RequestId >
  < HostId > ram - test - app . oss - cn - hangzhou . aliyuncs . com
</ HostId >
</ Error >
Error Status :
403
get Failed !
[ admin @ NGIS - CWWF344M01 C / home / admin / oss_test ]
$./ osscmd put test . txt oss :// ram - test - app / test . txt
-- host = oss - cn - hangzhou . aliyuncs . com - i o0hue *****
Frogv - k OmVwFJ03qc T0 ***** Fh0Ypg3p0K nA
100 % Error Headers :
[(' content - length ', ' 229 '), (' server ', ' AliyunOSS '), ('
connection ', ' keep - alive '), (' x - oss - request - id ', '
564A94E5B1 119B445B9F 8C3A '), (' date ', ' Tue , 17 Nov 2015
02 : 45 : 57 GMT '), (' content - type ', ' applicatio n / xml
')]
Error Body :
<? xml version =" 1 . 0 " encoding =" UTF - 8 "? >
< Error >
  < Code > AccessDeni ed </ Code >
  < Message > AccessDeni ed </ Message >
  < RequestId > 564A94E5B1 119B445B9F 8C3A </ RequestId >
  < HostId > ram - test - app . oss - cn - hangzhou . aliyuncs . com
</ HostId >
</ Error >
```

```
Error    Status :
403
put     Failed !
```

Without access permission, access attempts using the RAM user account `ram_test_app` are failed.

Use temporary authorization for downloads

Now, we use STS to download files. To make it simple to understand, the entered policy and the role policy are the same. The expiration time is set to 3600s, and the app user here is `usr001`. The steps are as follows:

1. Use STS to obtain a temporary credential.

```
[ admin @ NGIS - CWWF344M01 C / home / admin / oss_test ]
$ python ./ sts . py AssumeRole RoleArn = acs : ram ::
1894xxxxxx 722283 : role / ramtestapp readonly RoleSessio
nName = usr001 Policy = '{" Version ":" 1 ", " Statement ": [{"
Effect ":" Allow ", " Action ":" oss : ListObject s ", " oss :
GetObject "], " Resource ":" [ " acs : oss : *: *: ram - test - app ", "
acs : oss : *: *: ram - test - app /*"] ]}]' -- id = o0hue *****
Frogv -- secret = 0mVwFJ03qc T0 ***** Fh0Ypg3p0K nA
https :// sts . aliyuncs . com /? SignatureV ersion = 1 . 0
& Format = JSON & Timestamp = 2015 - 11 - 17T03 % 3A07 % 3A25Z
& RoleArn = acs % 3Aram % 3A % 3A1894xxxx xx722283 % 3Arole %
2Framtesta ppreadonly & RoleSessio nName = usr001 & AccessKeyI
d = o0hu ***** 3Frogv & Policy =% 7B % 22Version % 22 % 3A % 221
% 22 % 2C % 22Statemen t % 22 % 3A % 5B % 7B % 22Effect % 22 % 3A
% 22Allow % 22 % 2C % 22Action % 22 % 3A % 5B % 22oss % 3AListObje
cts % 22 % 2C % 22oss % 3AGetObjec t % 22 % 5D % 2C % 22Resource
% 22 % 3A % 5B % 22acs % 3Aoss % 3A % 2A % 3A % 2A % 3Aram - test
- app % 22 % 2C % 22acs % 3Aoss % 3A % 2A % 3A % 2A % 3Aram - test
- app % 2F % 2A % 22 % 5D % 7D % 5D % 7D & SignatureM ethod =
HMAC - SHA1 & Version = 2015 - 04 - 01 & Signature = bshxPZpwRJ
v5ch3SjaBi XLodwq0 % 3D & Action = AssumeRole & SignatureN once =
53e1be9c - 8cd8 - 11e5 - 9b86 - 008cfa5e49 38
{
  " AssumedRol eUser ": {
    " Arn ": " acs : ram :: 1894xxxxxx 722283 : role / ramtestapp
readonly / usr001 ",
    " AssumedRol eId ": " 3174463476 57426289 : usr001 "
  },
  " Credential s ": {
    " AccessKeyI d ": " STS . 3mQEbNf ***** wa180Le ",
    " AccessKeyS ecret ": " B1w7rCbR4d zGwNYJ ***** 3PiPqKZ3gj
QhAxb6mB ",
    " Expiration ": " 2015 - 11 - 17T04 : 07 : 25Z ",
    " SecurityTo ken ": " CAESvAMIAR KAASQQ ***** 7683CglhdG
sv2 / di8uI + X ***** DxM5FTd0fp 5wpPK / 7UctYH2MJ ///
c4yMN1PUCc EHI1zppCIN mpDG2XeNA3 OS16JwS6ES mI50sHyWBm
sYkCJW15gX nfhz / OK + mSp1bYxlfB 33qfgCFe97 Ijeuj8RMgq
Fx0Hny2BzG hhTVFMuM21 RRWJOZnR5Y zllT3dhMTg wTGUiEjMxN
zQ0NjM0NzY 1NzQyNjI40 SoGdXNyMDA xMJTrgJ2RK joGUnNhTUQ
1QpsBCGExG pUBCGVBbG ***** CgxBY3Rpb2 5FcXVhbHMS BkFjdGlvbh
ogCg9vc3M6 TGlzdE9iam VjdHMKDW9z czpHZXRPyM plY3QSUgo0
UmVzb3VyY2 VFcXVhbHMS CFJlc291cm NlGjYKGGFj czpvc3M6Kj
oqOnJhbS10 ZXN0LWFwcA oaYWNzOm9z czoq ***** FtLXRlc3Qt
YXBwLypkED E40TQxODk3 Njk3MjIyOD NSBTI2ODQy Wg9Bc3N1bW
```

```
VkUm9sZVVz ZXJgAGoSMz E3NDQ2MzQ3 NjU3NDI2Mj g5chJyYW10
ZXN0YXBwcm VhZG9ubHk ="
  },
  "RequestId ": " 8C009F64 - F19D - 4EC1 - A3AD - 7A718CD0B4 9B "
}
```

2. Use the temporary credential to download files. Here sts_token is the SecurityToken returned by the STS.

```
[ admin @ NGIS - CWWF344M01 C / home / admin / oss_test ]
$ ./ osscmd get oss :// ram - test - app / test . txt test .
txt -- host = oss - cn - hangzhou . aliyuncs . com - i STS .
3mQEbnf ***** wa180Le - k B1w7rCbR4d zGwNYJ ***** 3PiPqKZ3gj
QhAxb6mB -- sts_token = CAESvAMIAR KAASQQ ***** 7683CglhdG sv2
/ di8uI + X ***** DxM5FTd0fp 5wpPK / 7UctYH2MJ /// c4yMN1PUCc
EHIlzpPCIN mpDG2XeNA3 OS16JwS6ES mI50sHyWBm sYkCJW15gX nfHz
/ OK + mSp1bYxlfB 33qfgCFe97 Ijeuj8RMgq Fx0Hny2BzG hhTVFMuM21
RRWJ0ZnR5Y zL1T3dhMTg wTGuiEjMxN zQ0NjM0NzY 1NzQyNjI40
SoGdXNyMDA xMJTrgJ2RK joGUUnhTUQ 1QpsBCgExG pUBCgVBbG *****
CgxBY3Rpb2 5FcXVhbHMS BkFjdGlvbh ogCg9vc3M6 TGlzdE9iam
VjdHMKDW9z czpHZXRPyM pLY3QSUGo0 UmVzb3VyY2 VFcXVhbHMS
CFJlc291cm NLGjYKGGFj czpvc3M6Kj oqOnJhbS10 ZXN0LWFwCA
oaYWNzOm9z czoq ***** FtLXRlc3Qt YXBwLypKED E40TQxODk3
Njk3MjIyOD NSBTI2ODQy Wg9Bc3N1bW VkUm9sZVVz ZXJgAGoSMz
E3NDQ2MzQ3 NjU3NDI2Mj g5chJyYW10 ZXN0YXBwcm VhZG9ubHk =
100 % The object test . txt is downloaded to test .
txt , please check .
0 . 061 ( s ) elapsed
```

3. As you can see, we can use the temporary credentials to download the file. Next, we will test if we can use them to upload a file.

```
[ admin @ NGIS - CWWF344M01 C / home / admin / oss_test ]
$ ./ osscmd put test . txt oss :// ram - test - app / test .
txt -- host = oss - cn - hangzhou . aliyuncs . com - i STS .
3mQEbnf ***** wa180Le - k B1w7rCbR4d zGwNYJ ***** 3PiPqKZ3gj
QhAxb6mB -- sts_token = CAESvAMIAR KAASQQ ***** 7683CglhdG sv2
/ di8uI + X ***** DxM5FTd0fp 5wpPK / 7UctYH2MJ /// c4yMN1PUCc
EHIlzpPCIN mpDG2XeNA3 OS16JwS6ES mI50sHyWBm sYkCJW15gX nfHz
/ OK + mSp1bYxlfB 33qfgCFe97 Ijeuj8RMgq Fx0Hny2BzG hhTVFMuM21
RRWJ0ZnR5Y zL1T3dhMTg wTGuiEjMxN zQ0NjM0NzY 1NzQyNjI40
SoGdXNyMDA xMJTrgJ2RK joGUUnhTUQ 1QpsBCgExG pUBCgVBbG *****
CgxBY3Rpb2 5FcXVhbHMS BkFjdGlvbh ogCg9vc3M6 TGlzdE9iam
VjdHMKDW9z czpHZXRPyM pLY3QSUGo0 UmVzb3VyY2 VFcXVhbHMS
CFJlc291cm NLGjYKGGFj czpvc3M6Kj oqOnJhbS10 ZXN0LWFwCA
oaYWNzOm9z czoq ***** FtLXRlc3Qt YXBwLypKED E40TQxODk3
Njk3MjIyOD NSBTI2ODQy Wg9Bc3N1bW VkUm9sZVVz ZXJgAGoSMz
E3NDQ2MzQ3 NjU3NDI2Mj g5chJyYW10 ZXN0YXBwcm VhZG9ubHk =
100 % Error Headers :
[(' content - length ', ' 254 '), (' server ', ' AliyunOSS '), ('
connection ', ' keep - alive '), (' x - oss - request - id ', '
564A9A2A17 90CF0F53C1 5C82 '), (' date ', ' Tue , 17 Nov
2015 03 : 08 : 26 GMT '), (' content - type ', ' applicatio n
/ xml ')]
Error Body :
<? xml version =" 1 . 0 " encoding =" UTF - 8 "? >
< Error >
  < Code > AccessDeni ed </ Code >
  < Message > Access denied by authorizer ' s policy .</
Message >
  < RequestId > 564A9A2A17 90CF0F53C1 5C82 </ RequestId >
```

```
< HostId > ram - test - app . oss - cn - hangzhou . aliyuncs .
com </ HostId >
</ Error >
Error Status :
403
put Failed !
```

The file upload is failed. This is because the assumed role only has download permission hence.

Use temporary authorization for uploads

Now, we will try to use STS to upload a file. The steps are as follows:

1. Obtain an STS temporary credential. The app user is usr001.

```
[ admin @ NGIS - CWWF344M01 C / home / admin / oss_test ]
$ python ./ sts . py AssumeRole RoleArn = acs : ram ::
1894xxxxxx 722283 : role / ramtestapp write RoleSessio nName
= usr001 Policy ='{" Version ":" 1 "," Statement ":[{" Effect ":"
Allow "," Action ":[{" oss : PutObject "," Resource ":[{" acs : oss
:***: ram - test - app / usr001 /*"}]}]}' -- id = o0hue ***** Frogv
-- secret = 0mVwFJ03qc T0 ***** Fh0Ypg3p0K nA
https :// sts . aliyuncs . com /? SignatureV ersion = 1 . 0
& Format = JSON & Timestamp = 2015 - 11 - 17T03 % 3A16 % 3A10Z
& RoleArn = acs % 3Aram % 3A % 3A1894xxxx xx722283 % 3Arole %
2Framtesta ppwrite & RoleSessio nName = usr001 & AccessKeyI d
= o0hu ***** 3Frogv & Policy =% 7B % 22Version % 22 % 3A % 221 %
22 % 2C % 22Statemen t % 22 % 3A % 5B % 7B % 22Effect % 22 % 3A %
22Allow % 22 % 2C % 22Action % 22 % 3A % 5B % 22oss % 3APutObjec
t % 22 % 5D % 2C % 22Resource % 22 % 3A % 5B % 22acs % 3Aoss % 3A
% 2A % 3A % 2A % 3Aram - test - app % 2Fusr001 % 2F % 2A % 22 % 5D
% 7D % 5D % 7D & SignatureM ethod = HMAC - SHA1 & Version = 2015
- 04 - 01 & Signature = Y00PUoL1Pr CqX4X6A3 % 2FJvgXuS6c % 3D &
Action = AssumeRole & SignatureN once = 8d0798a8 - 8cd9 - 11e5 -
9f49 - 008cfa5e49 38
{
  " AssumedRol eUser ": {
    " Arn ": " acs : ram :: 1894xxxxxx 722283 : role / ramtestapp
write / usr001 ",
    " AssumedRol eId ": " 3554078476 60029428 : usr001 "
  },
  " Credential s ": {
    " AccessKeyI d ": " STS . rtfx13 ***** NlIJlS4U ",
    " AccessKeyS ecret ": " 2fsaM8E2ma B2dn ***** wpsKTyK4aj
o7TxFr0zIM ",
    " Expiration ": " 2015 - 11 - 17T04 : 16 : 10Z ",
    " SecurityTo ken ": " CAESkwMIAR KAAUh3 / Uzcgl3 *****
y0IZjGewMp g31ITxCleB FUle0 / 3Sgpudid + GV s + 0lvu1vXJn *****
a8azKJKtzV 0oKSy + mwUrxSvUSR VDntrs78Cs NfWo0JUMJK jLIxdWnGi1
pgxJCBzNZ2 YV / 6ycTaZySSE 1V6kqQ7A + GPwY ***** LpdGhhTVFM
ucnRmeDEzR FlNVWJjTmx JSmxTNFuie jMINTQwnZg 0NzY2MDAYo
TQy0CoGdXN yMDAxMOPzo J2RKjoGUNn hTUQ1QnYKA TEacQoFQWx
sb3cSJwoMQ WN0aW9uRXF 1YWxzEgZBY 3Rpb24aDwo Nb3Nz0lB1d
E9iamVjdBI / Cg5SZXNvdX JjZUVxdWFs cxIIUmVzb3 VyY2UaIwoh
YWNz0m9zcz oq0io6cmFt LXRlc3Qt ***** VzcyjAwMS8q ShAx0Dk0MT
g5NzY5NzIy MjgzUgUyNj g0Ml0PQXNz dW1lZFJvbG VVc2VyYABq
EjMINTQwnZ g0NzY2MDAY OTQyOHIPcm FtdGVzdGFw cHdyaxRl "
  },
  " RequestId ": " 19407707 - 54B2 - 41AD - AAF0 - FE87E8870B 0D "
```

```
}
```

2. Let us test if we can use the credentials to upload and download.

```
[ admin @ NGIS - CWWF344M01 C / home / admin / oss_test ]
$./ osscmd get oss :// ram - test - app / test . txt test .
txt -- host = oss - cn - hangzhou . aliyuncs . com - i STS .
rtfx13 ***** NLIJLS4U - k 2fsaM8E2ma B2dn ***** wpsKTyK4aj
o7TxFr0zIM -- sts_token = CAESkwMIAR KAAUh3 / Uzcg13 *****
y0IZjGewMp g31ITxCleB FUle0 / 3Sgpubid + GVs + Olvu1vXJn *****
a8azKJKtzV 0oKSy + mwUrxSvUSR VDntrs78Cs NfWo0JUMJK jLIxdWnGi1
pgxJCBzNZ2 YV / 6ycTaZySSE 1V6kqQ7A + GPwY ***** LpdGhhTVFM
ucnRmeDEzR FLNVWJjTmx JSmxTNFUiE jM1NTQwNzg 0NzY2MDAY0
TQy0CoGdXN yMDAxM0Pzo J2RKjoGUnN hTUQ1QnYKA TEacQoFQWx
sb3cSJwoMQ WN0aw9uRXF 1YWxzEgZBY 3Rpb24aDwo Nb3Nz0lB1d
E9iamVjdBI / Cg5SZXNvdX JjZUVxdWFs cxIIUmVzb3 VyY2UaIwoh
YWNzOm9zcz oq0io6cmFt LXRLc3Qt ***** VzcjAwMS8q ShAxODk0MT
g5NzY5NzIy MjgzUgUyNj g0MloPQXNz dW1lZFJvbG VVc2VyYABq
EjM1NTQwNz g0NzY2MDAY OTQyOHIPcm FtdGVzdGFw cHdyaXRl
Error Headers :
[(' content - length ', ' 254 '), (' server ', ' AliyunOSS '), ('
connection ', ' keep - alive '), (' x - oss - request - id ', '
564A9C31FF FC811F24B6 E7E3 '), (' date ', ' Tue , 17 Nov
2015 03 : 17 : 05 GMT '), (' content - type ', ' applicatio n
/ xml ')]
Error Body :
<? xml version =" 1 . 0 " encoding =" UTF - 8 "? >
< Error >
  < Code > AccessDeni ed </ Code >
  < Message > Access denied by authorizer ' s policy .</
Message >
  < RequestId > 564A9C31FF FC811F24B6 E7E3 </ RequestId >
  < HostId > ram - test - app . oss - cn - hangzhou . aliyuncs .
com </ HostId >
</ Error >
Error Status :
403
get Failed !
[ admin @ NGIS - CWWF344M01 C / home / admin / oss_test ]
$./ osscmd put test . txt oss :// ram - test - app / test .
txt -- host = oss - cn - hangzhou . aliyuncs . com - i STS .
rtfx13 ***** NLIJLS4U - k 2fsaM8E2ma B2dn ***** wpsKTyK4aj
o7TxFr0zIM -- sts_token = CAESkwMIAR KAAUh3 / Uzcg13 *****
y0IZjGewMp g31ITxCleB FUle0 / 3Sgpubid + GVs + Olvu1vXJn *****
a8azKJKtzV 0oKSy + mwUrxSvUSR VDntrs78Cs NfWo0JUMJK jLIxdWnGi1
pgxJCBzNZ2 YV / 6ycTaZySSE 1V6kqQ7A + GPwY ***** LpdGhhTVFM
ucnRmeDEzR FLNVWJjTmx JSmxTNFUiE jM1NTQwNzg 0NzY2MDAY0
TQy0CoGdXN yMDAxM0Pzo J2RKjoGUnN hTUQ1QnYKA TEacQoFQWx
sb3cSJwoMQ WN0aw9uRXF 1YWxzEgZBY 3Rpb24aDwo Nb3Nz0lB1d
E9iamVjdBI / Cg5SZXNvdX JjZUVxdWFs cxIIUmVzb3 VyY2UaIwoh
YWNzOm9zcz oq0io6cmFt LXRLc3Qt ***** VzcjAwMS8q ShAxODk0MT
g5NzY5NzIy MjgzUgUyNj g0MloPQXNz dW1lZFJvbG VVc2VyYABq
EjM1NTQwNz g0NzY2MDAY OTQyOHIPcm FtdGVzdGFw cHdyaXRl
100 % Error Headers :
[(' content - length ', ' 254 '), (' server ', ' AliyunOSS '), ('
connection ', ' keep - alive '), (' x - oss - request - id ', '
564A9C3FB8 DE437A91B1 6772 '), (' date ', ' Tue , 17 Nov
2015 03 : 17 : 19 GMT '), (' content - type ', ' applicatio n
/ xml ')]
Error Body :
<? xml version =" 1 . 0 " encoding =" UTF - 8 "? >
< Error >
  < Code > AccessDeni ed </ Code >
```



```
< Message > Access denied by authorizer 's policy .</
Message >
  < RequestId > 564A9C3FB8 DE437A91B1 6772 </ RequestId >
  < HostId > ram - test - app . oss - cn - hangzhou . aliyuncs .
com </ HostId >
</ Error >
Error Status :
403
put Failed !
```

The test.txt upload fails. We have formatted the entered policy discussed at the beginning of this document, which is as follows:

```
{
  " Version ": " 1 ",
  " Statement ": [
    {
      " Effect ": " Allow ",
      " Action ": [
        " oss : PutObject "
      ],
      " Resource ": [
        " acs : oss :*:ram - test - app / usr001 /*"
      ]
    }
  ]
}
```

This policy indicates that users are only allowed to upload files like usr001/ to the ram-test-app bucket. If the app user is usr002, the policy can be changed to only allow for the uploading of files like usr002/. By setting different policies for different app users, we can isolate the storage space of different app users.

3. Retry the test and specify the upload destination as ram-test-app/usr001/test.txt.

```
[ admin @ NGIS - CWWF344M01 C / home / admin / oss_test ]
$ ./ osscmd put test . txt oss :// ram - test - app / usr001
/ test . txt -- host = oss - cn - hangzhou . aliyuncs . com -
i STS . rtfx13 ***** NLIJLS4U - k 2fsaM8E2ma B2dn *****
wpsKTyK4aj o7TxFr0zIM -- sts_token = CAESkwMIAR KAAUh3 / Uzcg13
***** y0IZjGewMp g31ITxCleB FU1e0 / 3Sgpudid + GVs + 0lvu1vXJn
***** a8azKJKtzV 0oKSy + mwUrxSvUSR VDntrs78Cs NfWo0JUMJK
jLIxdWnGi1 pgxJCBzNZ2 YV / 6ycTaZySSE 1V6kqQ7A + GPwY *****
LpdGhhTVFM ucnRmeDEzR FlNVWJjTmx JSmxTNFUie jM1NTQwNzg
0NzY2MDAY0 TQy0CoGdXN yMDAXMOPzo J2RKjoGUnN hTUQ1QnYKA
TEacQoFQWx sb3cSJwoMQ WN0aW9uRXF 1YWxzEgZBY 3Rpb24aDwo
Nb3Nz0lB1d E9iamVjdBI / Cg5SZXNvdX JjZUVxdWFs cxIIUmVzb3
VyY2UaIwoh YWNzOm9zcz oq0io6cmFt LXRlc3Qt ***** VzCjAwMS8q
ShAxODk0MT g5NzY5NzIy MjgzUgUyNj g0MloPQXNz dW1lZFJvbG
VVc2VyYABq EjM1NTQwNz g0NzY2MDAY OTQyOHIPcm FtdGVzdGFw
cHdyXRl
100 %
Object URL is : http :// ram - test - app . oss - cn -
hangzhou . aliyuncs . com / usr001 % 2Ftest . txt
Object abstract path is : oss :// ram - test - app / usr001
/ test . txt
ETag is " 946A0A1AC8 245696B9C6 A6F3594269 0B "
```

```
0 . 071 ( s ) elapsed
```

The upload is successful.

Summary

This section describes how to grant users temporary access authorization for OSS using STS. In typical mobile development scenarios, STS can be used to grant temporary authorizations to access OSS when different app users need to access the app. The temporary authorization can be configured with expiration time to greatly reduce the hazards caused by leaks. When obtaining temporary authorization, we can enter different authorization policies for different app users to restrict their access permissions. For example, to restrict the object paths accessible to users. This isolates the storage space of different app users.

21.1.3 FAQ about subaccount settings

How to create an STS temporary account and how to use it to access resources?

See [STS temporary access authorization](#).

Client or console logon error reported for an authorized sub-account

See [Why does a sub-account encounters an error of no operation permission for a bucket on the OSS console after it has been granted the bucket operation permission](#).

How to authorize a sub-account with the operation permission for a single bucket

See [How to assign the full operation permission for a specified bucket to a sub-account](#).

How to authorize a sub-account with the operation permission for a directory in a bucket

See [OSS directory authorization](#)

How to authorize a sub-account with the read-only permission for a bucket

See [Authorize a sub-user to list and read resources in a bucket](#).

Error upon an OSS SDK call: InvalidAccessKeyId

See [STS errors and troubleshooting](#).

Error upon an STS call: Access denied by authorizer' s policy

Detailed error information: ErrorCode: AccessDenied ErrorMessage: Access denied by authorizer' s policy.

Cause of the error:

- The temporary account has no access permission.
- The authorization policy specified for assuming the role of this temporary account does not assign the access permission to the account.

For more STS errors and the causes, see [OSS permission errors and troubleshooting](#).