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, consequent
ial, 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 contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, 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 this document 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.
# Document conventions

<table>
<thead>
<tr>
<th>Style</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>🔴</td>
<td>A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.</td>
<td>Danger: Resetting will result in the loss of user configuration data.</td>
</tr>
<tr>
<td>⚠️</td>
<td>A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.</td>
<td>Warning: Restarting will cause business interruption. About 10 minutes are required to restart an instance.</td>
</tr>
<tr>
<td>🚨</td>
<td>A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.</td>
<td>Notice: If the weight is set to 0, the server no longer receives new requests.</td>
</tr>
<tr>
<td>📝</td>
<td>A note indicates supplemental instructions, best practices, tips, and other content.</td>
<td>Note: You can use Ctrl + A to select all files.</td>
</tr>
<tr>
<td>&gt;</td>
<td>Closing angle brackets are used to indicate a multi-level menu cascade.</td>
<td>Click Settings &gt; Network &gt; Set network type.</td>
</tr>
<tr>
<td><strong>Bold</strong></td>
<td>Bold formatting is used for buttons, menus, page names, and other UI elements.</td>
<td>Click OK.</td>
</tr>
<tr>
<td><strong>Courier font</strong></td>
<td>Courier font is used for commands.</td>
<td>Run the cd /d C:/window command to enter the Windows system folder.</td>
</tr>
<tr>
<td><em>Italic</em></td>
<td>Italic formatting is used for parameters and variables.</td>
<td>bae log list --instanceid Instance_ID</td>
</tr>
<tr>
<td>[ ] or [a</td>
<td>b]</td>
<td>This format is used for an optional value, where only one item can be selected.</td>
</tr>
<tr>
<td>Style</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>{} or {a</td>
<td>b}</td>
<td>This format is used for a required value, where only one item can be selected.</td>
</tr>
</tbody>
</table>
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Legal disclaimer</td>
<td>I</td>
</tr>
<tr>
<td>Document conventions</td>
<td>I</td>
</tr>
<tr>
<td>1 Usage instructions</td>
<td>1</td>
</tr>
<tr>
<td>2 Basic concepts</td>
<td>2</td>
</tr>
<tr>
<td>3 Endpoint</td>
<td>7</td>
</tr>
<tr>
<td>3.1 Regions and endpoints</td>
<td>7</td>
</tr>
<tr>
<td>3.2 Endpoints</td>
<td>12</td>
</tr>
<tr>
<td>4 Storage classes</td>
<td>15</td>
</tr>
<tr>
<td>4.1 Introduction to storage classes</td>
<td>15</td>
</tr>
<tr>
<td>4.2 Convert between storage classes</td>
<td>20</td>
</tr>
<tr>
<td>4.3 Create and use Archive buckets</td>
<td>23</td>
</tr>
<tr>
<td>5 Access OSS</td>
<td>30</td>
</tr>
<tr>
<td>5.1 Quick start</td>
<td>30</td>
</tr>
<tr>
<td>5.2 OSS-based app development</td>
<td>31</td>
</tr>
<tr>
<td>6 Compliant retention policy</td>
<td>37</td>
</tr>
<tr>
<td>7 Disaster recovery</td>
<td>41</td>
</tr>
<tr>
<td>7.1 Redundant storage across zones</td>
<td>41</td>
</tr>
<tr>
<td>7.2 Cross-region replication</td>
<td>42</td>
</tr>
<tr>
<td>8 Buckets</td>
<td>45</td>
</tr>
<tr>
<td>8.1 Create a bucket</td>
<td>45</td>
</tr>
<tr>
<td>8.2 Set the ACL for a bucket</td>
<td>46</td>
</tr>
<tr>
<td>8.3 Obtain the region information of a bucket</td>
<td>48</td>
</tr>
<tr>
<td>8.4 View the bucket list</td>
<td>49</td>
</tr>
<tr>
<td>8.5 Enable the pay-by-requester mode</td>
<td>50</td>
</tr>
<tr>
<td>8.6 Bind a custom domain</td>
<td>52</td>
</tr>
<tr>
<td>8.7 Configure hotlink protection</td>
<td>54</td>
</tr>
<tr>
<td>8.8 Transfer acceleration</td>
<td>56</td>
</tr>
<tr>
<td>8.9 Set CORS rules</td>
<td>61</td>
</tr>
<tr>
<td>8.10 Bucket tagging</td>
<td>63</td>
</tr>
<tr>
<td>8.11 Delete a bucket</td>
<td>65</td>
</tr>
<tr>
<td>9 Objects</td>
<td>67</td>
</tr>
<tr>
<td>9.1 Upload files</td>
<td>67</td>
</tr>
<tr>
<td>9.1.1 Simple upload</td>
<td>67</td>
</tr>
<tr>
<td>9.1.2 Form upload</td>
<td>68</td>
</tr>
<tr>
<td>9.1.3 Multipart upload and resumable upload</td>
<td>72</td>
</tr>
<tr>
<td>9.1.4 Append upload</td>
<td>76</td>
</tr>
<tr>
<td>9.1.5 Authorized third-party upload</td>
<td>78</td>
</tr>
<tr>
<td>9.1.6 Upload callback</td>
<td>80</td>
</tr>
</tbody>
</table>
14 Manage logs
   14.1 Access logging
   14.2 Real-time log query
15 Data encryption
   15.1 Server-side encryption
16 Static website hosting
   16.1 Configure static website hosting
   16.2 Tutorial: Host a static website using a custom domain name
17 OSS sandbox
18 Monitoring service
   18.1 Monitoring service overview
   18.2 Monitoring service user guide
   18.3 Alert service
   18.4 Metric item reference
   18.5 Monitoring indicators reference
   18.6 Service monitoring, diagnosis, and troubleshooting
19 Cloud data processing
20 Event notification
21 Hide
   21.1 Access control
      21.1.1 Overview
      21.1.2 What is RAM and STS
      21.1.3 Access a bucket without using the primary account
      21.1.4 Read/Write permission separation
      21.1.5 STS temporary access authorization
      21.1.6 FAQ about subaccount settings
   21.2 Temporary authorized access
   21.3 CDN-based OSS acceleration
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:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer Guide</td>
<td>Describes the core concepts, functions, and provides methods (through console, API, or SDK) of using these functions.</td>
</tr>
<tr>
<td>Best Practices</td>
<td>Describes the application scenarios and configuration practices of OSS.</td>
</tr>
<tr>
<td>SDK Reference</td>
<td>Describes the SDK development, related parameters, and code samples based on major languages.</td>
</tr>
<tr>
<td>API Reference</td>
<td>Describes the RESTful API operations supported by OSS and provides related examples.</td>
</tr>
<tr>
<td>Console User Guide</td>
<td>Describes all operations supported by the OSS console.</td>
</tr>
<tr>
<td>Image Processing Guide</td>
<td>Describes various functions provided by Image Processing, such as format conversion, cropping, scaling, rotation, watermarks, and style encapsulation.</td>
</tr>
<tr>
<td>OSS migration tool</td>
<td>Describes the official migration tool that helps you migrate data from your local or third-party storage service to OSS.</td>
</tr>
</tbody>
</table>
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 #unique_12.

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

<table>
<thead>
<tr>
<th>Comparison item</th>
<th>OSS</th>
<th>File system</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data model</strong></td>
<td>OSS is a distributed object storage service that uses a key-value pair format.</td>
<td>The file system is a hierarchical tree structure of directories that contain files.</td>
</tr>
<tr>
<td><strong>Data retrieval</strong></td>
<td>Objects are retrieved based on unique object names (keys). 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.</td>
<td>Files are retrieved based on their locations in directories.</td>
</tr>
<tr>
<td><strong>Advantage</strong></td>
<td>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.</td>
<td>Folder operations such as renaming, moving, and deleting directories are quite easy, because data does not need to be copied and replaced.</td>
</tr>
</tbody>
</table>
Disadvantage | OSS | File system
--- | --- | ---
The stored objects cannot be modified directly. If you want to modify an object, you must upload the new object of the same name to replace the existing one. | 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.

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:

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

3.1 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.

<table>
<thead>
<tr>
<th>Region name</th>
<th>OSS region</th>
<th>Internet endpoint</th>
<th>Internet endpoint protocol</th>
<th>Intranet endpoint for ECS access</th>
<th>Intranet endpoint protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>China (Hangzhou)</td>
<td>oss-cn-hangzhou</td>
<td>oss-cn-hangzhou.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-cn-hangzhou-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Shanghai)</td>
<td>oss-cn-shanghai</td>
<td>oss-cn-shanghai.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-cn-shanghai-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Qingdao)</td>
<td>oss-cn-qingdao</td>
<td>oss-cn-qingdao.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-cn-qingdao-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Beijing)</td>
<td>oss-cn-beijing</td>
<td>oss-cn-beijing.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-cn-beijing-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Region name</td>
<td>OSS region</td>
<td>Internet endpoint</td>
<td>Internet endpoint protocol</td>
<td>Intranet endpoint for ECS access</td>
<td>Intranet endpoint protocol</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------</td>
<td>----------------------------</td>
<td>----------------------------</td>
<td>------------------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>China (Zhangjiakou)</td>
<td>oss-cn-zhangjiakou</td>
<td>oss-cn-zhangjiakou.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-cn-zhangjiakou-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Hohhot)</td>
<td>oss-cn-huhehaote</td>
<td>oss-cn-huhehaote.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-cn-huhehaote-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Shenzhen)</td>
<td>oss-cn-shenzhen</td>
<td>oss-cn-shenzhen.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-cn-shenzhen-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Chengdu)</td>
<td>oss-cn-chengdu</td>
<td>oss-cn-chengdu.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-cn-chengdu-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Hong Kong)</td>
<td>oss-cn-hongkong</td>
<td>oss-cn-hongkong.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-cn-hongkong-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>US (Silicon Valley)</td>
<td>oss-us-west-1</td>
<td>oss-us-west-1.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-us-west-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>US (Virginia)</td>
<td>oss-us-east-1</td>
<td>oss-us-east-1.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-us-east-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Region name</td>
<td>OSS region</td>
<td>Internet endpoint</td>
<td>Internet endpoint protocol</td>
<td>Intranet endpoint for ECS access</td>
<td>Intranet endpoint protocol</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------</td>
<td>----------------------------</td>
<td>---------------------------</td>
<td>---------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>Singapore</td>
<td>oss-ap-southeast-1</td>
<td>oss-ap-southeast-1.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-ap-southeast-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Australia (Sydney)</td>
<td>oss-ap-southeast-2</td>
<td>oss-ap-southeast-2.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-ap-southeast-2-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Malaysia (Kuala Lumpur)</td>
<td>oss-ap-southeast-3</td>
<td>oss-ap-southeast-3.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-ap-southeast-3-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Indonesia (Jakarta)</td>
<td>oss-ap-southeast-5</td>
<td>oss-ap-southeast-5.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-ap-southeast-5-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Japan (Tokyo)</td>
<td>oss-ap-northeast-1</td>
<td>oss-ap-northeast-1.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-ap-northeast-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>India (Mumbai)</td>
<td>oss-ap-south-1</td>
<td>oss-ap-south-1.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-ap-south-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Germany (Frankfurt)</td>
<td>oss-eu-central-1</td>
<td>oss-eu-central-1.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-eu-central-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Region name</td>
<td>OSS region</td>
<td>Internet endpoint</td>
<td>Internet endpoint protocol</td>
<td>Intranet endpoint for ECS access</td>
<td>Intranet endpoint protocol</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------</td>
<td>-----------------------</td>
<td>----------------------------</td>
<td>----------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>UK (London)</td>
<td>oss-eu-west-1</td>
<td>oss-eu-west-1.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-eu-west-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>UAE (Dubai)</td>
<td>oss-me-east-1</td>
<td>oss-me-east-1.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
<td>oss-me-east-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
</tbody>
</table>

**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.

<table>
<thead>
<tr>
<th>Region name</th>
<th>OSS region</th>
<th>Endpoint in VPC networks</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>China (Hangzhou)</td>
<td>oss-cn-hangzhou</td>
<td>oss-cn-hangzhou-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Region name</td>
<td>OSS region</td>
<td>Endpoint in VPC networks</td>
<td>Protocol</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>China (Shanghai)</td>
<td>oss-cn-shanghai</td>
<td>oss-cn-shanghai-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Qingdao)</td>
<td>oss-cn-qingdao</td>
<td>oss-cn-qingdao-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Beijing)</td>
<td>oss-cn-beijing</td>
<td>oss-cn-beijing-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Zhangjiakou)</td>
<td>oss-cn-zhangjiakou</td>
<td>oss-cn-zhangjiakou-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Hohhot)</td>
<td>oss-cn-huhehaote</td>
<td>oss-cn-huhehaote-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Shenzhen)</td>
<td>oss-cn-shenzhen</td>
<td>oss-cn-shenzhen-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Chengdu)</td>
<td>oss-cn-chengdu</td>
<td>oss-cn-chengdu-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>China (Hong Kong)</td>
<td>oss-cn-hongkong</td>
<td>oss-cn-hongkong-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>US (Silicon Valley)</td>
<td>oss-us-west-1</td>
<td>oss-us-west-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>US (Virginia)</td>
<td>oss-us-east-1</td>
<td>oss-us-east-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Singapore</td>
<td>oss-ap-southeast-1</td>
<td>oss-ap-southeast-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Australia (Sydney)</td>
<td>oss-ap-southeast-2</td>
<td>oss-ap-southeast-2-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
</tbody>
</table>
### 3.2 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

<table>
<thead>
<tr>
<th>Region name</th>
<th>OSS region</th>
<th>Endpoint in VPC networks</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Malaysia (Kuala Lumpur)</td>
<td>oss-ap-southeast-3</td>
<td>oss-ap-southeast-3-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Indonesia (Jakarta)</td>
<td>oss-ap-southeast-5</td>
<td>oss-ap-southeast-5-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Japan (Tokyo)</td>
<td>oss-ap-northeast-1</td>
<td>oss-ap-northeast-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>India (Mumbai)</td>
<td>oss-ap-south-1</td>
<td>oss-ap-south-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>Germany (Frankfurt)</td>
<td>oss-eu-central-1</td>
<td>oss-eu-central-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>UK (London)</td>
<td>oss-eu-west-1</td>
<td>oss-eu-west-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
<tr>
<td>UAE (Dubai)</td>
<td>oss-me-east-1</td>
<td>oss-me-east-1-internal.aliyuncs.com</td>
<td>HTTP and HTTPS</td>
</tr>
</tbody>
</table>

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?
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:

  ```html
  <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.

  ```java
  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.
  - `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:

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

**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 bucket 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.
4 Storage classes

4.1 Introduction to storage classes

OSS provides three storage classes: Standard, Infrequent Access (IA), and Archive, covering 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 ideal for storing various images that are used for social networking and sharing, and storing data for audio and video applications, large websites, and big data analytics.

- Supported data redundancy mechanisms
  - Locally redundant storage (LRS): uses the data redundancy storage mechanism to store data of each object on multiple devices in the same region, ensuring data durability and availability in case of hardware failure.
  - Zone-redundant storage (ZRS): uses the multi-zone mechanism to distribute user data across three zones within the same region. Even if one zone becomes unavailable, the data will still be accessible.

  Note:
  Before zone-redundant storage was released, locally redundant storage was the storage mechanism for standard storage.

- Key features
  - Provides 99.9999999999% (twelve 9's) data durability (designed for).
  - Provides 99.995% service availability (designed for).
  - Delivers high-throughput and low-latency access performance.
  - Supports HTTPS-based transmission.
  - Supports Image Processing (IMG).
IA

OSS IA storage is suitable for storing long-lived, but less frequently accessed data (an average of once or twice per month). IA storage offers a storage unit price that is lower than Standard storage, and is suitable for long-term backup of various mobile apps, smart device data, and enterprise data. It also supports real-time data access. Objects of the IA storage class have a minimum storage duration. Fees occur 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 fees.

- Supported data redundancy mechanisms
  - Locally redundant storage (LRS): uses the data redundancy storage mechanism to store data of each object on multiple devices in the same region, ensuring data durability and availability in case of hardware failure.
  
  **Note:**
  Before zone-redundant storage was released, locally redundant storage was the storage mechanism for IA storage.
  
  - Zone-redundant storage (ZRS): uses the multi-zone mechanism to distribute user data across three zones within the same region. Even if one zone becomes unavailable, the data will still be accessible.

- Key features
  - Provides 99.9999999999% (twelve 9's) data durability (designed for).
  - Provides 99.995% service availability (designed for).
  - Supports real-time access.
  - Supports HTTPS-based transmission.
  - Supports IMG.
  - 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 long-term storage (at least half a year) of data that is infrequently accessed. The data may take up to one minute to restore before it can be read. This storage option is suitable for data such as archival data, medical images, scientific materials, and video footage. Objects of the Archive storage class have a minimum
storage period. Fees occur 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 fees.

The Archive storage class has the following features:

- Provides 99.9999999999% (eleven 9's) data durability (designed for).
- Provides 99.99% service availability (designed for).
- Takes about one minute to restore the data from the frozen state to the readable state.
- Supports HTTPS-based transmission.
- Supports IMG, but data needs to be restored first.
- Requires a minimum storage duration and minimum billable size.

### Comparison of storage classes

<table>
<thead>
<tr>
<th>Item</th>
<th>Standard</th>
<th>IA</th>
<th>Archive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supported data redundancy mechanisms</td>
<td>LRS and ZRS</td>
<td>LRS and ZRS</td>
<td>N/A</td>
</tr>
<tr>
<td>Data durability (designed for)</td>
<td>99.9999999999% (twelve 9's)</td>
<td>99.9999999999% (twelve 9's)</td>
<td>99.9999999999% (eleven 9's)</td>
</tr>
<tr>
<td>Service availability (designed for)</td>
<td>99.995%</td>
<td>99.995%</td>
<td>99.99% (restored data)</td>
</tr>
<tr>
<td>Minimum billable size of objects</td>
<td>Actual size of objects</td>
<td>64 KB</td>
<td>64 KB</td>
</tr>
<tr>
<td>Minimum storage duration</td>
<td>No minimum storage duration</td>
<td>30 days</td>
<td>60 days</td>
</tr>
<tr>
<td>Data retrieval fee</td>
<td>No data retrieval fee</td>
<td>Based on the size of retrieved data. Unit: GB</td>
<td>Based on the size of restored data. Unit: GB</td>
</tr>
<tr>
<td>Data access</td>
<td>Real-time access with low latency (within milliseconds)</td>
<td>Real-time access with low latency (within milliseconds)</td>
<td>One minute after data is restored</td>
</tr>
<tr>
<td>IMG</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported after data is restored</td>
</tr>
</tbody>
</table>

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

Supported API operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Standard</th>
<th>IA</th>
<th>Archive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bucket creation, deletion, and query</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PutBucket</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>GetBucket</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>DeleteBucket</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Bucket ACL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PutBucketAcl</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>GetBucketAcl</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Bucket logging</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>PutBucketLogging</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>GetBucketLogging</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Bucket static website hosting</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PutBucketWebsite</td>
<td>Supported</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>GetBucketWebsite</td>
<td>Supported</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>Bucket hotlink protection</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>PutBucketReferer</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>GetBucketReferer</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Bucket lifecycle</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PutBucketLifecycle</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported for data deletion only</td>
</tr>
<tr>
<td>GetBucketLifecycle</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>DeleteBucketLifecycle</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Cross-region replication</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported for restored data</td>
</tr>
<tr>
<td>PutBucketReplication</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Operation</td>
<td>Standard</td>
<td>IA</td>
<td>Archive</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------</td>
<td>------</td>
<td>---------</td>
</tr>
<tr>
<td>Cross-origin resource sharing (CORS)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PutBucketcors</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>GetBucketcors</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>DeleteBucketcors</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Object operations</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PutObject</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>PutObjectACL</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>GetObject</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported after data is restored</td>
</tr>
<tr>
<td>GetObjectACL</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>GetObjectMeta</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>HeadObject</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>CopyObject</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>OptionObject</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>DeleteObject</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>DeleteMultipleObjects</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>PostObject</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>PutSymlink</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>GetSymlink</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>RestoreObject</td>
<td>Not supported</td>
<td>Not supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Multipart operations</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>InitiateMultipartUpload</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>UploadPart</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>UploadPartCopy</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>CompleteMultipartUpload</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>AbortMultipartUpload</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>ListMultipartUpload</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
</tbody>
</table>
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*.

**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.
• APIs or SDKs

The following code uses the Java SDK as an example:

```java
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

- 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:

```java
ObjectMetadata objectMetadata = ossClient.getObjectMetadata(bucketName, key);
// Check whether the storage class of the object is Archive.
StorageClass storageClass = objectMetadata.getObjectStorageClass();
if (storageClass == StorageClass.Archive) {
```

![Cannot preview this file, please wait until it's restored.](image)
// 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 \textit{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

- \textit{Set up direct data transfer for mobile apps}
- \textit{Set up data callback for mobile apps}
- \textit{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:

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 policy

OSS supports the Write Once Read Many (WORM) strategy that prevents an object from being deleted or overwritten for a period of time. This strategy can be used in environments that are subject to regulations of the U.S. Securities and Exchange Commission (SEC) and Financial Industry Regulatory Authority, Inc. (FINRA).

Note:

- The compliant retention policy function is in the beta testing phase. To use this function, contact After-sales technical support.
- In OSS, you can only configure a compliant retention policy for buckets.
- OSS is the only cloud service in China that has passed the audit and certification of Cohasset Associates and can meet specific requirements for electronic data storage. OSS configured with compliant retention policies can be used in environments that are subject to regulations such as SEC Rule 17a-4(f), CFTC Rule 1.31(c)-(d), and FINRA Rule 4511(c). For more information, see OSS Cohasset Assessment Report.

OSS provides strong compliant policies. You can configure time-based compliant retention policies for buckets. After a compliant retention policy is locked, you can read objects from or upload objects to buckets. However, no one can delete the protected objects or compliant retention policies within the retention period. You can delete objects only after their retention period ends. The WORM strategy is suitable for industries such as the financing, insurance, health care, and security. OSS allows you to build a "compliant bucket in the cloud."

Note:

During the protection period of objects in buckets, you can Set lifecycle rules to convert the storage class to minimize costs while ensuring compliance.

Implementation mode

Console: Set a compliant retention policy
Rules

You can add only one time-based compliant retention policy that has a protection period ranging from one day to 70 years.

Assume that you created a bucket named examplebucket on June 1, 2013, and uploaded the file1.txt, file2.txt, and file3.txt objects to the bucket at different points in time. Then, you created a compliant retention policy that has a protection period of five years on July 1, 2014. The following table describes the dates on which these objects were uploaded and are to expire.

<table>
<thead>
<tr>
<th>Object</th>
<th>Upload date</th>
<th>Expiration date</th>
</tr>
</thead>
<tbody>
<tr>
<td>file1.txt</td>
<td>June 1, 2013</td>
<td>May 31, 2018</td>
</tr>
<tr>
<td>file2.txt</td>
<td>July 1, 2014</td>
<td>June 30, 2019</td>
</tr>
<tr>
<td>file3.txt</td>
<td>September 30, 2018</td>
<td>September 29, 2023</td>
</tr>
</tbody>
</table>

- Implementation rule

After a time-based compliant retention policy is created for a bucket, the policy is in the InProgress state by default, which is valid for 24 hours. Within the validity period, the retention policy protects the resources in the bucket.

- Within 24 hours after the compliant retention policy is enabled: If the retention policy is not locked, the bucket owner and authorized users can delete this policy. If the retention policy is locked, no one can delete this policy or shorten the protection period. The protection period can only be prolonged.

- 24 hours after the compliant retention policy is enabled: If the retention policy is not locked, the policy becomes invalid.

- If you attempt to delete or modify data in the protected bucket, the OSS API returns 409 FileImmutable.

- Deletion rules

- A time-based compliant retention policy is a metadata property of a bucket. When a bucket is deleted, the compliant retention policy and ACL of the
bucket are also deleted. To delete the retention policy of a bucket, you can delete the bucket when the bucket is empty.

- Within 24 hours after a retention policy is created for a bucket, if the retention policy is still not locked, the bucket owner and authorized users can delete the policy.
- If a bucket stores objects that are within the protection period, you cannot delete the bucket or its compliant retention policy.

FAQ

• What are the advantages of a compliant retention policy?

  The compliant retention policy provides compliant storage for your data. No one can delete or modify data within the protection period of the compliant retention policy. However, data protected through RAM policies and bucket policies may be modified and deleted.

• What scenarios are suitable for a compliant retention policy?

  You can use the compliant retention policy when you want to store infrequently accessed important data that cannot be modified or deleted. Such data includes medical records, technical documents, and contracts. You can store these archive objects in a specified bucket and enable the compliant retention policy for the bucket.

• Does a compliant retention policy apply to individual objects?

  Only buckets can have the compliant retention policy enabled. Folders and individual objects cannot have the compliant retention policy enabled.

• How to delete a bucket protected by a compliant retention policy?

  - If no object is stored in a bucket, the bucket can be deleted.
  - If the bucket stores objects that are no longer within the protection period, the bucket cannot be deleted. In this case, you can delete all objects in the bucket before you delete the bucket.
  - If the bucket stores objects that are within the protection period, the bucket cannot be deleted.
• Are objects within the protection period of the compliant retention policy retained if my account has overdue OSS payments?

Alibaba Cloud retains data in an overdue account based on the terms and conditions of your contract.

• Can an authorized RAM user configure a compliant retention policy?

All API operations related to the compliant retention policy are available. Related API operations support RAM policies. RAM users authorized through RAM policies can create or delete the compliant retention policy through the console, API, or SDK.
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 99.9999999999% data durability (design for) and 99.995% service availability (design for).

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.

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.

<table>
<thead>
<tr>
<th>Index</th>
<th>Standard</th>
<th>IA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data durability (designed for)</td>
<td>99.99999999999% (twelve 9's)</td>
<td>99.99999999999% (twelve 9's)</td>
</tr>
<tr>
<td>Service availability (designed for)</td>
<td>99.995%</td>
<td>99.995%</td>
</tr>
<tr>
<td>Storage measurement unit</td>
<td>Actual object size</td>
<td>64 KB</td>
</tr>
<tr>
<td>Minimum storage period</td>
<td>No limit</td>
<td>30 days</td>
</tr>
<tr>
<td>Data retrieval fee</td>
<td>No fee is charged for data retrieval.</td>
<td>Charged based on the size of retrieved data (GB).</td>
</tr>
<tr>
<td>Data access</td>
<td>Real-time access with latency of milliseconds</td>
<td>Real-time access with latency of milliseconds</td>
</tr>
<tr>
<td>Images processing</td>
<td>Supported</td>
<td>Supported</td>
</tr>
</tbody>
</table>
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 scenarios and limits of cross-region replication.

Cross-region replication enables the automatic and asynchronous replication of objects across buckets in different OSS data centers (regions). It synchronizes operations such as the creation, overwrite, and deletion of objects from the source bucket to the destination bucket. This feature meets users' data replication requirements, and provides an ideal cross-region disaster recovery method for buckets. Objects in the destination bucket are exact replicas of objects in the source bucket. They have the same object name, object metadata, and content, such as the creation time, owner, user metadata, object ACL, and object content.

Note:

- After cross-region replication is enabled, cross-region traffic is generated when you replicate objects between buckets in the primary and secondary regions. OSS charges fees for the traffic occurred during cross-region replication. Each time an object is synchronized, OSS counts the number of requests and calculates the charges on a Pay-As-You-Go basis.
- For more information about regions that support the cross-region replication feature and the pricing, see the "Cross Region Replication" section in Object Storage Service Pricing.

Scenarios

Cross-region replication can be configured for a wide range of scenarios:

- Compliance requirements: Although OSS stores multiple replicas of each object on physical disks, the replicas must be stored at a certain distance from each other to comply with regulations. Cross-region replication allows you to
replicate data between geographically distant OSS data centers to satisfy these compliance requirements.

- Minimum latency: You have users who are located in two geographical locations. To minimize latency when accessing objects, you can maintain replicas of the objects in OSS data centers that are geographically closer to users.
- Data backup and disaster recovery: You have high requirements for data security and availability, and want to explicitly maintain replicas of all written data in a second data center. If one OSS data center is damaged in a catastrophic event such as earthquake or tsunami, you can use backup data from the other data center.
- Data replication: For business reasons, you may need to migrate data from one OSS data center 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 replicas in these regions.

Implementation modes

<table>
<thead>
<tr>
<th>Implementation mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#unique_39</td>
<td>A user-friendly and intuitive Web application</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
</tbody>
</table>

Instructions

Cross-region replication supports the synchronization between buckets that have different names. If two buckets are in different regions, you can use this feature to synchronize the data from the source bucket to the destination bucket in real time. This feature now offers the following capabilities:

- Real-time data synchronization: This feature monitors data addition, deletion, and modification in real time and synchronizes these changes to the destination bucket. For objects no more than 2 MB in size, data is synchronized within a few minutes to ensure data consistency between the source and the destination.
- Historical data migration: This feature synchronizes historical data from the source bucket to the destination bucket, providing two identical data replicas.
Real-time display of synchronization progress: This feature displays the last synchronization time for real-time data synchronization and the percentage of synchronization for historical data migration.

Easy configuration: The OSS console provides easy-to-use GUIs for configuration management.

Mutual synchronization: You can configure data synchronization between Bucket A and Bucket B to enable mutual data synchronization.

Limits

You can still perform operations on buckets that are being synchronized. However, an object that is replicated from the source bucket may overwrite an object that has the same name in the destination bucket. Exercise caution when you synchronize data.

Cross-region replication is an asynchronous process. Based on the size of the data, it usually takes several minutes to several hours to copy data from the source bucket to the destination bucket.

Cross-region synchronization applies only when no synchronization to or from a third bucket is enabled for the two buckets to be synchronized. For example, if you synchronize data from Bucket A to Bucket B, you are not allowed to synchronize data from Bucket A to Bucket C, unless you delete the configuration to synchronize data from Bucket A to Bucket B. Similarly, if you synchronize data from Bucket A to Bucket B, you are not allowed to synchronize data from Bucket C to Bucket B.

Synchronization is supported only between two buckets in different regions. You cannot synchronize data between buckets in the same region.

Currently, the bucket cross-region replication feature is available only between regions in Mainland China and between US (Virginia) and US (Silicon Valley).
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 #unique_43.

Operating methods

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

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.
<table>
<thead>
<tr>
<th>ACL</th>
<th>Description</th>
<th>Access control</th>
</tr>
</thead>
</table>
| public-read-write | The public-read-write permission. | Anyone (including anonymous users) can perform read and write operations on the objects in the bucket.  
**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. |
| public-read   | The public-read permission.        | 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.  
**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. |
| 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**

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
</tbody>
</table>
### Operating methods

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ossbrowser</td>
<td>Graphical tool, which is easy to operate</td>
</tr>
<tr>
<td>ossutil</td>
<td>Command-line tool, which delivers good performance</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
<tr>
<td>Node.js SDK</td>
<td></td>
</tr>
<tr>
<td>Ruby SDK</td>
<td></td>
</tr>
</tbody>
</table>

#### 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](#).
### 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Console</strong></td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td><strong>ossbrowser</strong></td>
<td>Graphical tool, which is easy to operate</td>
</tr>
<tr>
<td><strong>ossutil</strong></td>
<td>Command-line tool, which delivers good performance</td>
</tr>
<tr>
<td><strong>Java SDK</strong></td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td><strong>Python SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>PHP SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Go SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>C SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>.NET SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Node.js SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Ruby SDK</strong></td>
<td></td>
</tr>
</tbody>
</table>
### 8.5 Enable the pay-by-requester mode

When 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 requests and traffic instead of the bucket owner. The bucket owner always pays the cost for storing data. You can enable this feature to share data without having to pay for the request and traffic costs by yourself.

#### Implementation modes

<table>
<thead>
<tr>
<th>Implementation mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Console</strong></td>
<td>A user-friendly and intuitive Web application</td>
</tr>
<tr>
<td><strong>Java SDK</strong></td>
<td>SDK demos for various programming languages</td>
</tr>
<tr>
<td><strong>Python SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Go SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>C++ SDK</strong></td>
<td></td>
</tr>
</tbody>
</table>

#### Use cases

- Share large datasets, such as postal 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 for which they are expected to pay the request and traffic costs. The configuration procedure is as follows:

  1. Enable the pay-by-requester mode for the bucket. For more information, see 
     #unique_70.

  2. Configure bucket policy to authorize the RAM user accounts of your customers' Alibaba Cloud accounts to access your bucket. For more information, see 
     #unique_75.
- 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. Configure bucket policy to authorize the RAM user accounts of your partners' Alibaba Cloud accounts to access your bucket. For more information, see Tutorial: Authorize a RAM user under another Alibaba Cloud account by adding a bucket policy.

Notice:
You must authorize the RAM user account of your customers' or partners' Alibaba Cloud accounts to access your bucket. However, ensure that you do not provide them with the AccessKey pair of your RAM user account. If your customers or partners use your RAM user account to access your bucket, you will be charged for requests and traffic because you are still the requester.

Request methods

- Access from anonymous users is not allowed

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

If requesters assume 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 request header

If you enable the pay-by-requester mode for a bucket, requesters must include the x-oss-request-payer:requester request header in the PUT, POST, GET, or HEAD request. This field indicates that requesters understand that they will be charged for requests and data downloads. Otherwise, requesters cannot be authenticated.

The bucket owner does not need to include the x-oss-request-payer request header when attempting to access their own buckets. In this case, the bucket owner pays for such requests and traffic.
Billing details

When the pay-by-requester mode is enabled, requesters pay for one or more of the following billing items based on their request content: outbound traffic over the public network, back-to-origin traffic, and traffic generated when making API operation calls, performing Image Processing (IMG), taking video snapshots, and retrieving IA objects or archives. The bucket owner pays other fees such as storage, object tagging, and transfer acceleration fees. However, requests fail (where HTTP status code 403 is returned) in the following cases and the bucket owner will be charged:

- The requester does not include the x-oss-request-payer request header in the 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.

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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Console</strong></td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td><strong>PHP SDK</strong></td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td><strong>Node.js SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Browser.js SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Ruby SDK</strong></td>
<td></td>
</tr>
</tbody>
</table>

### 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
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.

<table>
<thead>
<tr>
<th>Access process</th>
<th>Before user A binds the custom domain</th>
<th>After user A binds the custom domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. DNS resolves the IP address of user A's server from the request.</td>
<td>2. DNS resolves the OSS endpoint abc-img.oss-cn-hangzhou.aliyuncs.com from the request.</td>
<td>2. DNS resolves the OSS endpoint abc-img.oss-cn-hangzhou.aliyuncs.com from the request.</td>
</tr>
<tr>
<td>3. The user accesses the image logo.png on user A's server.</td>
<td>3. The user accesses the image logo.png in the OSS bucket abc-img.</td>
<td>3. The user accesses the image logo.png in the OSS bucket abc-img.</td>
</tr>
</tbody>
</table>

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 #unique_86.

8.7 Configure hotlink 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
<tr>
<td>Node.js SDK</td>
<td></td>
</tr>
<tr>
<td>Ruby SDK</td>
<td></td>
</tr>
</tbody>
</table>

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 FAQs for hotlinking protection, see OSS hotlink protection (referer) errors and troubleshooting.

8.8 Transfer acceleration

OSS can use transfer acceleration to help your enterprise reach its full potential across all Alibaba Cloud regions. OSS uses data centers distributed around the globe to implement transfer acceleration. After a data transfer request is sent, it is resolved and then smartly routed over an optimal network path and protocol to the data center where your bucket is located. Transfer acceleration allows customers to
access OSS more quickly to improve customer experience and make business closer to customers.

Note:

- Additional fees are charged based on traffic when you use transfer acceleration. For more information, visit #unique_100/unique_100_Connect_42_section_dxe_pq2_8bw.
- This feature is available for all regions except UAE (Dubai).

Scenarios

OSS transfer acceleration is used to accelerate access and improve customer experience.

- A forum whose members are from around the globe has data stored in OSS. Upload or download speeds vary from region to region, compromising access experience. In this case, you can enable transfer acceleration to allow international members to transfer data over an optimal network. This way, access is accelerated and members from different regions can have better access experience.

- OSS cannot make full use of all available bandwidth over the Internet due to TCP-based limits. You can enable transfer acceleration to maximize available bandwidth to accelerate transfers of objects.

Instructions

After transfer acceleration is enabled for a bucket, the bucket has two endpoints that provide services over the public network.

- Accelerate endpoint: oss-accelerate.aliyuncs.com. When you want to accelerate access for data transfers, you can use this address to access OSS. For example, a bucket named test is located in US (Silicon Valley). An object named 123.jpg is stored in the root directory. When you access resources stored in OSS through the browser, use the accelerate endpoint http://test.oss-accelerate.aliyuncs.com/123.jpg.

- Regular endpoint: The format is oss-region.aliyuncs.com. You can use this address to access OSS without transfer acceleration. For more information about regular endpoints, see Regions and endpoints.
Additional fees are incurred when you use transfer acceleration. To minimize costs, we recommend that you use this feature as follows:

- You can use accelerate endpoints to upload or download objects when access experience is required.
- You can use regular endpoints to perform operations such as configuring basic features for buckets and deleting objects when access experience is not required.
- To perform operations related to resource consumption such as Image Processing (IMG) and SelectObject, contact After-sales technical support to obtain a more suitable solution.
- If a bucket and an ECS instance are located within the same region and you want to access OSS from the ECS instance, we recommend that you use the OSS endpoint for access over the internal network. For example, a bucket and an ECS instance are located in China (Shanghai). We recommend that you use oss-cn-shanghai-internal.aliyuncs.com to access OSS from the ECS instance.

You can compare the access speeds of different regions between the accelerate and regular endpoints as instructed in The Comparison of OSS Direct Data Transfer and Accelerated Data Transfer in Different Regions.

Implementation modes

Enable OSS transfer acceleration as instructed in #unique_101. After transfer acceleration is enabled, you can use any of the following methods to implement transfer acceleration for access to data stored in OSS:
• Browser

When you use the browser to access data stored in OSS, replace the endpoint in the object URL with an accelerate endpoint. For example, replace https://test.oss-cn-shenzhen.aliyuncs.com/myphoto.jpg with https://test.oss-accelerate.aliyuncs.com/myphoto.jpg. If the object ACL is Private, include signature information.

Note:
To accelerate access to a bucket that has transfer acceleration and enabled and a custom domain name bound, configure CNAME to map your custom domain name to the accelerate endpoint. When you configure CNAME, specify the custom domain name for host records, set the CNAME value to an accelerate endpoint such as test.oss-accelerate.aliyuncs.com. For more information about how to configure CNAME for Alibaba Cloud domain names, see Attach a CDN acceleration domain name.

• ossutil

When you use ossutil to access data stored in OSS, replace the endpoint in the configuration file with an accelerate endpoint. Alternatively, you can add -e oss-accelerate.aliyuncs.com to each command to perform an operation. For more information about how to configure the endpoint when you use ossutil, see ossutil.

• ossbrowser

When you use ossbrowser to access data stored in OSS, set the endpoint to Customize, and specify an accelerate endpoint. For more information about how to configure the endpoint when you use ossbrowser, see ossbrowser.

• SDK

To access OSS through SDKs that use different languages, set the endpoint to an accelerate endpoint. The following code provides an example of how to use the OSS Java SDK to perform simple upload:

```java
// This example uses China (Hangzhou) as the endpoint. Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-accelerate.aliyuncs.com";
// Security risks may arise if you log on with the AccessKey pair of an Alibaba Cloud account because the account has permissions on all API operations in OSS. We recommend that you log on with a RAM user account to call API operations or perform routine operations and maintenance. To create a RAM user account, log on to https://ram.console.aliyun.com.
```
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>"

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId, accessKeySecret);

// Upload your local file. <yourLocalFile> consists of the local file path, file name, and file extension, such as /users/local/myfile.txt.
ossClient.putObject("<yourBucketName>", "<yourObjectName>", new File("<yourLocalFile>"));

// Close your OSSClient.
ossClient.shutdown();

For more information about SDK examples, see #unique_103.

Precautions

- After you enable or disable transfer acceleration, it takes about 30 minutes for configurations to take effect.
- The maximum bandwidth can reach 1 Gbit/s when you use an accelerate endpoint to access a bucket that has transfer acceleration enabled.
- After transfer acceleration is enabled, you must use the accelerate endpoint to accelerate access.
- When your tool has Endpoint set to an accelerate endpoint, you can perform operations only on the bucket that has transfer acceleration enabled.
- After transfer acceleration is enabled, other endpoints provided by OSS can be accessed properly. You can switch between endpoints at any time.
- Transfer acceleration applies for acceleration of access over the Internet.
- To secure data transfers, the acceleration logic of actual transfer acceleration uses HTTPS for data transfers as required. Therefore, if you use HTTPS for access, the actual accelerate endpoint-based URL uses HTTPS. If you use HTTP for access, the actual acceleration endpoint-based URL may be changed to HTTPS.

Billing

If you enable transfer acceleration and the accelerate endpoint is used to access your bucket, OSS charges additional fees. Assume that an accelerate endpoint is used to download data of 1 GB from a bucket that has transfer acceleration enabled. OSS charges fees based on traffic of 1 GB for transfer acceleration and outbound traffic over the public network of 1 GB. For more information about prices, visit the OSS pricing page.
To view transfer acceleration fees incurred when you use accelerate endpoints, go to Billing Management in Alibaba Cloud Management Console. The following table describes the billing items in detail.

<table>
<thead>
<tr>
<th>Billing item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccM2MOut</td>
<td>Accelerated downloads between regions inside Mainland China</td>
</tr>
<tr>
<td>AccM2MIn</td>
<td>Accelerated uploads between regions inside Mainland China</td>
</tr>
<tr>
<td>AccO2MOut</td>
<td>Accelerated downloads from regions outside Mainland China to buckets located inside Mainland China</td>
</tr>
<tr>
<td>AccO2MIn</td>
<td>Accelerated uploads from regions outside Mainland China to buckets located inside Mainland China</td>
</tr>
<tr>
<td>AccM2OOut</td>
<td>Accelerated downloads from regions inside Mainland China to buckets located outside Mainland China</td>
</tr>
<tr>
<td>AccM2OIn</td>
<td>Accelerated uploads from regions inside Mainland China to buckets located outside Mainland China</td>
</tr>
<tr>
<td>AccO2OOut</td>
<td>Accelerated downloads between regions outside Mainland China</td>
</tr>
<tr>
<td>AccO2OIn</td>
<td>Accelerated uploads between regions outside Mainland China</td>
</tr>
</tbody>
</table>

### 8.9 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
</tbody>
</table>

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`.

8.10 Bucket tagging

OSS allows you to configure bucket tagging to classify and manage buckets. For example, you can configure this feature to list only buckets that have specific tags. Bucket tagging uses a key-value pair to identify buckets. You can manage multiple buckets that have specific tags.
Only the bucket owner and authorized users can configure tagging for the bucket. If other users attempt to configure tagging for the bucket, 403 Forbidden is returned with error code AccessDenied.

You can configure a maximum of 20 tags for a bucket.

The tag key is required. The tag key can be a maximum of 64 bytes in length and cannot start with http://, https://, or Aliyun.

The tag value is optional. The tag value can be a maximum of 128 bytes in length.

The key and value of the tag must be encoded in UTF-8.

### Implementation modes

<table>
<thead>
<tr>
<th>Implementation mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
</tbody>
</table>

### Instructions

After you add tags to buckets, you can manage multiple buckets that have the same tag. For example, you can list buckets that have the same tag and authorize RAM users to manage buckets that have the same tag.

- Lists buckets that have specific tags

  You can list buckets that have specific tags. For more information, see the following SDK demos:

  - Java SDK
  - Python SDK
  - Go SDK

- Authorize RAM users to manage buckets that have specific tags

  When you have a large number of buckets, you can classify buckets based on tags and use the RAM policy to authorize specific users to manage buckets that have specific tags. For example, you can authorize user A to list buckets that have the tag configuration of keytest=valuetest. The RAM policy is as follows:

```json
{
  "Version": "1",
  "Statement": [
    {
      "Action": [
```


8.11 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Console</strong></td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td>ossbrowser</td>
<td>Graphical tool, which is easy to operate</td>
</tr>
<tr>
<td>ossutil</td>
<td>Command-line tool, which delivers good performance</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
<tr>
<td>Android SDK</td>
<td></td>
</tr>
<tr>
<td>iOS SDK</td>
<td></td>
</tr>
<tr>
<td>Operating method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Node.js SDK</td>
<td></td>
</tr>
<tr>
<td>Ruby SDK</td>
<td></td>
</tr>
</tbody>
</table>
9 Objects

9.1 Upload files

9.1.1 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td>ossbrowser</td>
<td>Graphical tool, which is easy to operate</td>
</tr>
<tr>
<td>ossutil</td>
<td>Command-line tool, which delivers good performance</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
<tr>
<td>Android SDK</td>
<td></td>
</tr>
<tr>
<td>iOS SDK</td>
<td></td>
</tr>
<tr>
<td>Node.js SDK</td>
<td></td>
</tr>
<tr>
<td>Browser.js SDK</td>
<td></td>
</tr>
</tbody>
</table>
### 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](#).

### 9.1.2 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.

<table>
<thead>
<tr>
<th>Other upload methods</th>
<th>Form upload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access process</td>
<td></td>
</tr>
<tr>
<td>1. A website user sends a request to upload a resume.</td>
<td>1. A website user sends a request to upload a resume.</td>
</tr>
<tr>
<td>2. The website server responds with a resume upload page.</td>
<td>2. The website server responds with a resume upload page.</td>
</tr>
<tr>
<td>3. The resume is uploaded to the website server.</td>
<td>3. The resume is uploaded to OSS.</td>
</tr>
<tr>
<td>4. The website server uploads the resume to OSS.</td>
<td></td>
</tr>
</tbody>
</table>

- 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.

```python
#coding=utf8
import md5
import hashlib
import base64
import hmac
from optparse import OptionParser
def convert_base64(input):
    return base64.b64encode(input)
def get_sign_policy(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, 104857600]
        ]
    }"
    print("policy: %s" % policy)
    #2 Use Base64 to encode the policy string.
    base64policy = convert_base64(policy)
    print("base64_encode_policy: %s" % base64policy)
    #3 Use the AccessKey Secret of OSS to add a signature to the encoded policy.
    signature = get_sign_policy(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">
```
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=ZQNJzf4QJRkrH4 --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.

9.1.3 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ossutil</code></td>
<td>Command-line tool, which delivers good performance</td>
</tr>
<tr>
<td><code>Java SDK</code></td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td><code>Python SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>PHP SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>Go SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>C SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>.NET SDK</code></td>
<td></td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Java SDK</code></td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td><code>Python SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>Go SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>C SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>.NET SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>Android SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>iOS SDK</code></td>
<td></td>
</tr>
</tbody>
</table>

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

9.1.4 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
<tr>
<td>Android SDK</td>
<td></td>
</tr>
<tr>
<td>iOS SDK</td>
<td></td>
</tr>
</tbody>
</table>

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.

9.1.5 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
</tbody>
</table>

- 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.
9.1.6 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.

![Diagram showing the process of upload callback]

**Note:**
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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
</tbody>
</table>

### Reference

- *Upload callback errors and troubleshooting*
- *Direct data transfer on Web clients and upload callback*
- *Set up upload callback for mobile applications*
9.1.7 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.

```python
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_cfg = LiveChannelInfo(target = LiveChannelInfoTarget())
channel = bucket.create_live_channel(channel_name, channel_cfg)
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=LGaxxxxxxHjKWg6&playlistName=test-channel
• 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=%2FAvRo7FTss1InBKgw7Gz%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.
9.2 Download files

9.2.1 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Console</strong></td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td><strong>osssbrowser</strong></td>
<td>Graphical tool, which is easy to operate</td>
</tr>
<tr>
<td><strong>ossutil</strong></td>
<td>Command-line tool, which delivers good performance</td>
</tr>
<tr>
<td><strong>Java SDK</strong></td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td><strong>Python SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>PHP SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Go SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>C SDK</strong></td>
<td></td>
</tr>
<tr>
<td>Operating method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Android SDK</td>
<td></td>
</tr>
<tr>
<td>iOS SDK</td>
<td></td>
</tr>
<tr>
<td>Node.js SDK</td>
<td></td>
</tr>
<tr>
<td>Browser.js SDK</td>
<td></td>
</tr>
<tr>
<td>Ruby SDK</td>
<td></td>
</tr>
</tbody>
</table>

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.

9.2.2 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>Android SDK</td>
<td></td>
</tr>
<tr>
<td>iOS SDK</td>
<td></td>
</tr>
</tbody>
</table>

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.

9.2.3 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>?OSSAccessKeyId=<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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
<tr>
<td>Android SDK</td>
<td></td>
</tr>
<tr>
<td>iOS SDK</td>
<td></td>
</tr>
</tbody>
</table>

Best practices

RAM and STS User Guide

9.2.4 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 #unique_211.

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).

9.3 Manage files

9.3.1 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**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache-Control</td>
<td>The Webpage caching behavior when the object is downloaded.</td>
</tr>
<tr>
<td>Content-Disposition</td>
<td>The name of the object during the download.</td>
</tr>
<tr>
<td>Content-Encoding</td>
<td>The content encoding format of the object during the download.</td>
</tr>
<tr>
<td>Content-Language</td>
<td>The content language encoding when the object is downloaded.</td>
</tr>
<tr>
<td>Expires</td>
<td>The time the object expires.</td>
</tr>
<tr>
<td>Content-Length</td>
<td>The size of the object.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>The type of the object.</td>
</tr>
<tr>
<td>Last-Modified</td>
<td>The time the object is last modified.</td>
</tr>
</tbody>
</table>
• 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td>ossbrowser</td>
<td>Graphical tool, which is easy to operate</td>
</tr>
<tr>
<td>ossutil</td>
<td>Command-line tool, which delivers good performance</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
<tr>
<td>Android SDK</td>
<td></td>
</tr>
</tbody>
</table>

Reference

You can also add Object Meta in the following operations:

• Simple upload
• Multipart upload and resumable upload
• Append upload
• Copy objects

9.3.2 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delimiter</td>
<td>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.</td>
</tr>
<tr>
<td>Marker</td>
<td>The starting position of the object list. Objects are sorted in alphabetical order and those objects following this marker are listed.</td>
</tr>
<tr>
<td>MaxKeys</td>
<td>The maximum number of objects that are returned. The default value is 100, and the maximum value is 1000.</td>
</tr>
<tr>
<td>Prefix</td>
<td>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.</td>
</tr>
</tbody>
</table>

Operating methods

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Web application that lists the objects in a bucket on the Files tab of the bucket overview page</td>
</tr>
<tr>
<td>ossbrowser</td>
<td>Graphical tool, which is easy to operate</td>
</tr>
<tr>
<td>ossutil</td>
<td>Command-line tool, which delivers good performance</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
</tbody>
</table>
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:

<table>
<thead>
<tr>
<th>Object</th>
<th>Directory</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object D</td>
<td>A/Object C</td>
<td></td>
</tr>
<tr>
<td>Directory A/Object D</td>
<td>B/Object B</td>
<td></td>
</tr>
<tr>
<td>Directory A/Directory B/Directory C/Object A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Directory A/Directory C/Object A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Directory A/Directory D/Object B</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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
<?xml version="1.0" encoding="UTF-8"?>
<ListBucketResult>
  <Name>oss-sample</Name>
  <Prefix></Prefix>
  <Marker></Marker>
  <MaxKeys>1000</MaxKeys>
  <Delimiter>/</Delimiter>
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Key>Object D</Key>
    <LastModified>2015-11-06T10:07:11.000Z</LastModified>
    <ETag>"8110930DA5E04B1ED5D84D6CC4DC9080"</ETag>
    <Type>Normal</Type>
    <Size>3340</Size>
    <StorageClass>Standard</StorageClass>
    <Owner>
      <ID>oss</ID>
      <DisplayName>oss</DisplayName>
    </Owner>
  </Contents>
  <CommonPrefixes>
    <Prefix>Directory A/</Prefix>
  </CommonPrefixes>
  <CommonPrefixes>
    <Prefix>Directory B/</Prefix>
  </CommonPrefixes>
</ListBucketResult>
```

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
<?xml version="1.0" encoding="UTF-8"?>
<ListBucketResult>
  <Name>oss-sample</Name>
  <Prefix>Directory A/</Prefix>
  <Marker></Marker>
  <MaxKeys>1000</MaxKeys>
  <Delimiter>/</Delimiter>
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Key>Directory A/Object C</Key>
    <LastModified>2015-11-06T09:36:00.000Z</LastModified>
    <ETag>"B026324C6904B2A9CB4B88D6D61C81D1"</ETag>
  </Contents>
</ListBucketResult>
```
In the preceding response:


9.3.3 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

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

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.

### 9.3.4 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Console</code></td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td><code>ossbrowser</code></td>
<td>Graphical tool, which is easy to operate</td>
</tr>
<tr>
<td><code>ossutil</code></td>
<td>Command-line tool, which delivers good performance</td>
</tr>
<tr>
<td><code>Java SDK</code></td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td><code>Python SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>PHP SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>Go SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>C SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>.NET SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>Android SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>Node.js SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>Browser.js SDK</code></td>
<td></td>
</tr>
<tr>
<td><code>Ruby SDK</code></td>
<td></td>
</tr>
</tbody>
</table>
9.3.5 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**

![Mirroring diagram]

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_object.jpg.
object.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:

<table>
<thead>
<tr>
<th>Content-Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Encoding</td>
</tr>
<tr>
<td>Content-Disposition</td>
</tr>
<tr>
<td>Cache-Control</td>
</tr>
<tr>
<td>Expires</td>
</tr>
<tr>
<td>Content-Language</td>
</tr>
<tr>
<td>Access-Control-Allow-Origin</td>
</tr>
</tbody>
</table>

- 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.

9.3.6 SelectObject

The SelectObject API operation is a log analysis tool that can be used in conjunction with other big data products. This topic describes how to use the Python SDK and Java SDK to call the SelectObject API operation in the preceding scenarios.

Introduction

Object Storage Service (OSS) is a secure and highly reliable cloud storage service based on the Apsara system for low-cost, general-purpose online storage of large amounts of data. It also supports RESTful API operations and automatic 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. Hadoop 3.0 is supported on OSS. You can directly read and process data in OSS when you run services such as Spark, Hive, and Presto in E-MapReduce or use Alibaba Cloud services such as MaxCompute, HybridDB for MySQL, and recently launched Data Lake Analytics.

However, the GetObject API operation supported by OSS only allows big data platforms to download all OSS data locally for analysis and filtering. As a result, bandwidth and client resources are wasted in many query scenarios.

To solve this problem, OSS provides the SelectObject API operation. SelectObject allows OSS to preliminarily filter data by using conditions and projections provided by big data platforms. As a result, only the relevant data is returned. This way, the client consumes fewer bandwidth resources and less data is processed to maximize CPU and memory resources utilization. This makes OSS-based data warehousing and analysis a highly attractive option.

SelectObject is currently supported by Java and Python SDKs, and will soon be supported by SDKs in other languages. SelectObject supports UTF-8 encoded CSV and JSON objects. Supported CSV objects and CSV-like objects such as TSV objects must conform to RFC 4180. You can customize row and column delimiters and quote characters in CSV objects. SelectObject supports Standard and Infrequent Access (IA) objects. Archives must be restored before you can perform SelectObject
Object Storage Service Developer Guide / 9 Objects

on them. SelectObject also supports objects that are fully managed and encrypted by OSS or encrypted with CMKs managed by KMS.

SelectObject supports JSON objects that are 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 row delimiters. However, the JSON object itself may not be valid. SelectObject supports typical delimiters such as \n and \r\n. You do not need to specify the delimiters.

- Supported SQL syntax

  - SQL statement: SELECT FROM WHERE
  - Data types: string, int64, double64, decimal128, timestamp, and Boolean
  - Operator-based operations: operations based on logical operators (AND, OR, and NOT), arithmetic operators (+, -, ×, /, and %), comparison operators (>\, =\, <\, ≥\, ≤\, and !=\), and string operators (LIKE and ||)

- Multipart query

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

- Data types

In OSS, the default data type of CSV objects is string. You can use the CAST function to convert the data into other types. For example, the following SQL statement converts 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, SelectObject allows you to implicitly convert the data type in a WHERE clause. For example, the following SQL statement converts 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 example

```python
import os
import oss2

def select_call_back(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 either of the following endpoints:
# # 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 operation.
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_call_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_call_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\n"]}
filename = 'python_select.json'
# Upload a JSON DOCUMENT object.
bucket.put_object(key, content)
select_json_params = {'Json_Type': '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 = {'Json_Type': '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_json_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_object(key)

**Details of SelectObject in Python**

- **select_object**

  - The following code provides an example of how to set parameters for select_object:

    ```python
def select_object(self, key, sql, progress_callback=None,
                   select_json_params):
    pass
```
The preceding sample code is used to execute an SQL statement on the specified object 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 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 of the GetObject API operation. For example, you can configure the bytes field in the request header to specify the range to query in a CSV object.

- The following table describes the parameters supported by the select_params parameter.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Json_Type       | ■ By default, the object is a CSV object if this parameter is not specified.  
■ If this parameter is set to DOCUMENT, the object is a JSON DOCUMENT object.  
■ If this parameter is set to LINES, the object is a JSON LINES object. |
| CsvHeaderInfo   | The header information of the CSV object. Valid values: None, Ignore, and Use.  
■ None: indicates that this object does not have header information.  
■ Ignore: indicates that this object has header information, which is not used in the SQL statement.  
■ Use: indicates that this object has header information, and the column names in the header information are used in the SQL statement. |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CommentCharacter</td>
<td>The comment character in the CSV object. You can specify only one character to start comments with. The default value is None, indicating that no comment characters are allowed.</td>
</tr>
<tr>
<td>RecordDelimiter</td>
<td>The row delimiter in the CSV object. This parameter value can be only one or two characters in length. Default value: \n</td>
</tr>
<tr>
<td>OutputRecordDelimiter</td>
<td>The row delimiter in the output result of the select_object operation. Default value: \n</td>
</tr>
<tr>
<td>FieldDelimiter</td>
<td>The column delimiter in the CSV object. This parameter value can be only one character in length. Default value: comma (,)</td>
</tr>
<tr>
<td>OutputFieldDelimiter</td>
<td>The column delimiter in the output result of the select_object operation. Default value: comma (,)</td>
</tr>
<tr>
<td>QuoteCharacter</td>
<td>The quote character for the columns in the CSV object. This parameter value can be only one character in length. Default value: double quotation marks (&quot;). Row and column delimiters enclosed in quotation marks are processed as normal characters.</td>
</tr>
<tr>
<td>SplitRange</td>
<td>The split range in multipart query. This parameter value is a closed interval in (start, end) format, indicating that splits from start# to end# are queried.</td>
</tr>
<tr>
<td>LineRange</td>
<td>The row range in multipart query. This parameter value is a closed interval in (start, end) format, indicating that rows from start# to end# are queried.</td>
</tr>
<tr>
<td>CompressionType</td>
<td>The compression type. Valid value: GZIP. Default value: None.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>KeepAllColumns</td>
<td>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. Example: The columns in the CSV object are 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 can directly use the code that is used to process GetObject to process SelectObject without modifications.</td>
</tr>
</tbody>
</table>
| OutputRawData       | ■ If this parameter is set to True, the select_object operation returns the original data. A timeout error may occur if it takes too long to return the data.  
■ If this parameter is set to False, the output data is encapsulated in frames. Default value: False. |
<p>| 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 only applies to CSV objects.                              |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SkipPartialDataRecord</td>
<td>■ 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.</td>
</tr>
<tr>
<td></td>
<td>■ If this parameter is set to False, columns without data are left empty in the output result.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>A row includes the columns firstname, lastname, and age. The SQL statement is <code>select _1, _4 from ossobject</code>. If this parameter is set to True, this row is skipped. If this parameter is set to False, firstname,\n is returned.</td>
</tr>
<tr>
<td>MaxSkippedRecordsAllowed</td>
<td>The maximum number of skipped rows. The default value is 0, indicating that an error is returned if a row is skipped.</td>
</tr>
<tr>
<td>ParseJsonNumberAsString</td>
<td>If this parameter is set to True, all numbers in the JSON object are parsed as strings. If this parameter is set to False, all numbers in the JSON object are parsed as integers or floating-point numbers. Default value: False.</td>
</tr>
<tr>
<td></td>
<td>High-precision floating-point numbers in a JSON object suffer loss of accuracy when they are parsed as floating-point numbers. To ensure precision, you can set this parameter to True and use the CAST function to convert the parsed data into the decimal type.</td>
</tr>
</tbody>
</table>

**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. This function is blocking 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 request.
processed by the OSS server in a timely manner. This way, the client does not need to wait until all results have been returned.

- **select_object_to_file**

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

The preceding code example is used to execute an SQL statement on the specified object 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**

  - **Syntax of select_meta_params**

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

The preceding code example is used to create Select Meta for a specified object or obtain Select Meta from a specified 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 already been created for an object, this function does not recreate Select Meta unless the value of the OverwriteIfExists parameter is set to true.

To create Select Meta for an object, you must completely scan the object.

- **Parameters supported by select_meta_params**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Json_Type</td>
<td>By default, the object is a CSV object if this parameter is not specified. If this parameter is specified, the parameter value must be LINES, indicating that the object is a JSON LINES object. This operation does not apply to JSON DOCUMENT objects.</td>
</tr>
<tr>
<td>RecordDelimiter</td>
<td>The row delimiter in the CSV object.</td>
</tr>
<tr>
<td>FieldDelimiter</td>
<td>The column delimiter in the CSV object.</td>
</tr>
<tr>
<td>QuoteCharacter</td>
<td>The quote character in the CSV object. Row and column delimiters enclosed in quote characters are processed as normal characters.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CompressionType</td>
<td>The compression type. If this parameter is specified, the parameter value must be None.</td>
</tr>
<tr>
<td>OverwriteIfExists</td>
<td>Indicates whether the created Select Meta overwrites the original Select Meta. You do not need to set this parameter in most cases.</td>
</tr>
</tbody>
</table>

- Returned result of create_select_object_meta: A GetSelectObjectMetaResult object is returned and 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

```java
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.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 metadata and select object.
 */

class MultipartSelector implements Callable<Integer> {
    private OSS client;
    private String bucket;
    private String key;
    private int start;
    private int end;
    private String sql;
```
```java
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)
            .withInputSerialization(
                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 = "<access Key Id that can do assume role>";
    public static final String serviceAccessKeySecret = "<access key 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 ClientException {
    try {
        // Create an AcsClient instance for sending API requests.
        if (this.client == null) {
            IClientProfile profile = DefaultProfile.getProfile(
                REGION_CN_HANGZHOU, accessKeyId, accessKeySecret);
            this.client = new DefaultAcsClient(profile);
        }
        // Create an AssumeRoleRequest 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 operation include RoleArn, RoleSessionName, Policy, and DurationSeconds.
    // You must obtain the value of the RoleArn parameter from 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 that you want to issue tokens to.
    // However, you must pay attention to the length and naming conventions of the RoleSessionName parameter. It can contain only letters, digits, 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 = "{" +
        "\"Version\": \"1\", \n" +
        "\"Statement\": [" +
        "{" +
            "\"Action\": \"oss:*\", \n" +
            "\"Resource\": [" +
            "\"acs:oss:*:*:*\"]\n" +
        "]\n" +
        "\"principal\": {" +
        "\"AWS\": [" +
        "\"acs:arn:cn-hangzhou:oss:*:*:*\"]\n" +
    "}\n" +
    "\"id\": \"acs:arn:cn-hangzhou:oss:*:*:*\"\n" +
};

    // Create an AcsClient instance for sending API requests.
    IClientProfile profile = DefaultProfile.getProfile(
        REGION_CN_HANGZHOU, accessKeyId, accessKeySecret);
    this.client = new DefaultAcsClient(profile);
    // Create an AssumeRoleRequest 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;
}
"effect": "Allow",
"]
"};

// Set the protocol type to HTTPS.
ProtocolType protocolType = ProtocolType.HTTPS;
try {
    final AssumeRoleResponse response = assumeRole(serviceKeyId, 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 tp 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 {
        CredentialsProvider provider = GetCredentialProvider();
        credential = provider.getCredentials();
        expireTime = Calendar.getInstance();
        expireTime.add(Calendar.SECOND, (int) DurationSeconds - 60);
    } catch (IOException e) {
        throw new InvalidCredentialsException("Unable tp get the temporary AK:" + e);
    }
}

public class SelectObjectSample {
    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 permissions to access OSS.
    private static String recordDelimiter = "\n";
    private static int threadCount = 10;
}
public static void main(String[] args) throws Exception {
    ClientBuilderConfiguration config = new ClientBuilderConfiguration();
    RoleCredentialProvider provider = new RoleCredentialProvider(roleArn);
    Credentials credentials = provider.getCredentials();
    //OSS client = new OSSClientBuilder().build(endpoint,
    accessKeyId, accessKeySecret, config);
    System.out.println("Id " + credentials.getAccessKeyId());
    System.out.println("Key " + credentials.getSecretAccessKey());
    System.out.println("Token " + credentials.getSecurityToken());
    OSS client = new OSSClientBuilder().build(endpoint,
    credentials.getAccessKeyId(), credentials.getSecretAccessKey(), credentials.
    getSecurityToken(), config);
    int totalSplits = 1;
    try {
        SelectObjectMetadata selectObjectMetadata = client.
        createSelectObjectMetadata(
            new CreateSelectObjectMetadataRequest(bucketName,
            key)
                .withInputSerialization(
                    new InputSerialization().
                        withCsvInputFormat(
                            new CSVFormat().
                                withHeader
                                Info(CSVFormat.Header.None).
                                withRecordDelimiter(recordDelimiter))));
        totalSplits = selectObjectMetadata.getCsvObjectMetadata().
        getSplits();
        System.out.println(selectObjectMetadata.getCsvObje
tMetadata().getTotalLines());
        System.out.println(totalSplits);
    } catch (Exception e) {
        e.printStackTrace();
    }
    String sql = "select count(*) from ossobject";
    ExecutorService executor = Executors.newFixedThreadPool(
    threadCount);
    long startTime = System.currentTimeMillis();
    List<Future<Integer>> list = new ArrayList<Future<Integer>>();
    int n = threadCount < totalSplits ? threadCount: totalSplits;
    for(int i = 0; i < n; i++) {
        int start = i * totalSplits/n;
        int end = i == n-1 ? totalSplits - 1 : (i+1)* totalSplits
        /n - 1;
        System.out.println("start:" + start + " end:" + end);
        Callable<Integer> task = new MultipartSelector(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.currentTimeMillis();
    System.out.println("total lines:" + totalLines);
System.out.printf("Total time %dms\n", (endTime - startTime));
}
}

SQL statement examples

- **SQL statement examples for CSV objects**

<table>
<thead>
<tr>
<th>Scenario</th>
<th>SQL statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return the first 10 rows.</td>
<td><code>select * from ossobject limit 10</code></td>
</tr>
<tr>
<td>Convert the values in the first and third columns to integers and return the integers of the first column that are greater than those of the third column.</td>
<td><code>select _1, _3 from ossobject where cast(_1 as int) &gt; cast(_3 as int)</code></td>
</tr>
<tr>
<td>Return the number of records in which the data in the first column starts with X. (Chinese characters specified after &quot;like&quot; must be UTF-8 encoded.)</td>
<td><code>select count(*) from ossobject where _1 like 'X%'</code></td>
</tr>
<tr>
<td>Return all records for 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 greater than 200.</td>
<td><code>select * from ossobject where _2 &gt; cast('2018-08-09 11:30:25' as timestamp) and _3 &gt; 200</code></td>
</tr>
<tr>
<td>Return the average value, sum, maximum value, and minimum value of the floating-point numbers in the second column.</td>
<td><code>select AVG(cast(_2 as double)), SUM(cast(_2 as double)), MAX(cast(_2 as double)), MIN(cast(_2 as double))</code></td>
</tr>
<tr>
<td>Return all records in which the strings concatenated by the data in the first and third columns start with Tom and end with Anderson.</td>
<td>`select * from ossobject where (_1</td>
</tr>
<tr>
<td>Return all records in which the data in the first column is divisible by 3.</td>
<td><code>select * from ossobject where (_1 % 3) == 0</code></td>
</tr>
<tr>
<td>Return all records in which the data in the first column ranges from 1995 to 2012.</td>
<td><code>select * from ossobject where _1 between 1995 and 2012</code></td>
</tr>
<tr>
<td>Return all records in which the data in the fifth column is N, M, G, or L.</td>
<td><code>select * from ossobject where _5 in ('N', 'M', 'G', 'L')</code></td>
</tr>
</tbody>
</table>
Object Storage Service

Developer Guide / 9 Objects

<table>
<thead>
<tr>
<th>Scenario</th>
<th>SQL statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return all records in which the product of the data in the second and third columns is greater than the sum of 100 and the data in the fifth column.</td>
<td>select * from ossobject where _2 * _3 &gt; _5 + 100</td>
</tr>
</tbody>
</table>

- 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
    }
  ],
  "children": [],
  "spouse": null
}
```

The following table describes the SQL statement examples.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>SQL statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return all records in which the value of age is 27.</td>
<td>select * from ossobject.contacts[*] s where s. age = 27</td>
</tr>
<tr>
<td>Return all home phone numbers.</td>
<td>select s.number from ossobject.contacts[<em>].phoneNumbers[</em>] s where s.type = &quot;home&quot;</td>
</tr>
<tr>
<td>Return all records in which the value of spouse is null.</td>
<td>select * from ossobject s where s.spouse is null</td>
</tr>
<tr>
<td>Scenario</td>
<td>SQL statement</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Return all records in which the value of</td>
<td>select * from ossobject s where s.children[0] is null</td>
</tr>
<tr>
<td>children is left empty.</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The preceding statement is used because an empty array cannot be specified in other ways.

**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 if 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 want to call SelectObject for the object, call it asynchronously before the query to shorten the time required for scanning.

  2. Specify the appropriate number of concurrencies (n) based on resources on the client. Divide the total number of splits by the concurrency n to obtain 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. Combine the results as needed.

- Use SelectObject for normal objects. We recommend that you do not use SelectObject to query multipart and appendable objects. The differences in their internal structures may deteriorate the query performance.

- When querying a JSON object, narrow the JSON path range in the FROM clause.

The following JSON object is used as an example.

```json
{
  "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 postal code starts with 10021, execute the following SQL statement:

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

Alternatively, execute the following SQL statement:

```sql
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 specific.

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

  If you want to calculate high-precision 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 parsed data into the decimal type.

  For example, the value of attribute a is 123456789.123456789. You can execute

  ```sql
  select s.a from ossobject s where cast(s.a as decimal) > 123456789.12345
  ```

  to maintain the accuracy of attribute a.

### 9.3.7 Object tagging

OSS allows you to use object tagging to classify objects. You can configure lifecycle rules for objects based on tags.

**Note:**

You are charged when using object tagging. For more information, see [OSS pricing](#).

Object tagging uses a key-value pair to identify objects. You can configure tags for objects when and after you upload the objects.

- A maximum of 10 tags can be set for each object. Tags associated with an object must have unique tag keys.
- A tag key can be a maximum of 128 bytes in length. Each tag value can be a maximum of 256 bytes in length.
- Keys and values are case-sensitive.
- The key and value of the tag can contain letters, digits, spaces, and special characters such as
  
  `+ - = . _ : /`

- Only the bucket owner and authorized users have the read and write permissions on object tags. These permissions are independent of object ACLs.

- During cross-region replication, object tags are also replicated to the destination bucket.

Scenarios

Object tags are not limited to folders. You can perform the following operations on multiple objects that have a specific tag:

- Configure lifecycle rules based on specific tags. For example, when you upload objects, you can configure tags for temporary objects that are periodically generated. After you configure lifecycle rules, you can delete these objects based on specific tags.

- Use RAM to grant permissions to access objects that have specific tags.

Implementation mode

<table>
<thead>
<tr>
<th>Implementation mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C++ SDK</td>
<td></td>
</tr>
</tbody>
</table>
Instructions

- **API operations related to object tagging**
  - `#unique_259`: Configures a tag for an object. If the target object already has a tag, the original tag is overwritten.
  - `#unique_260`: Reads tags of an object.
  - `#unique_261`: Deletes tags that are associated with an object.
  - `#unique_123`: You can use the `x-oss-tagging` request header to specify tags when you upload an object.
  - `#unique_156`: You can use the `x-oss-tagging` request header to specify tags when you initialize a multipart upload task.
  - `#unique_241`: You can use the `x-oss-tagging-directive` request header to specify whether to replicate tags of source objects. You can use the `x-oss-tagging` request header to specify tags of destination objects.
  - `#unique_192`: If you have permissions to read the object tags, the count of tags is included in the `x-oss-tagging-count` response header.
  - `#unique_262`: If you have permissions to read the object tags, the count of tags is included in the `x-oss-tagging-count` response header.

- **Permission description**

  Users, roles, or services that perform operations on tags must have the following permissions:
  
  - **GetObjectTagging**: The permission to obtain object tags. If you have this permission, you can view existing tags of objects.
  - **PutObjectTagging**: The permission to configure tags for objects. If you have this permission, you can configure tags for objects.
  - **DeleteObjectTagging**: The permission to delete object tags. If you have this permission, you can delete object tags.

Object tagging and lifecycle management

When you configure lifecycle rules, you can specify a filter to select a subset of objects to which the rule applies. You can specify a filter based on the key name prefixes, object tags, or both.

- If you set tag conditions, the rule only applies to objects that meet the tag key and value conditions.
• If you set object key prefixes and multiple object tags in one lifecycle rule, the
  rule only applies to objects that meet the object key prefixes and object tags.

Examples:

```xml
<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>
```

In the preceding rules,

• Objects that are prefixed with rule1 and whose tag configurations are xx=1 and
  yy=2 are deleted after 30 days.
• The storage class of objects that are prefixed with rule2 and whose tag configurat
  ion is xx=1 is converted to Archive after 60 days.

Note:
For more information, see Manage lifecycle rules.

Object tagging and RAM policies

You can authorize RAM users to manage object tags. You can also authorize RAM
users to manage objects that have specific tags.

• Authorize RAM users to manage object tags

You can authorize RAM users to manage tags of all or specific objects. If user
A is authorized to set object tagging to allow=yes, this user can add the tag
configuration of allow=yes for objects. An example of the RAM policy is as
follows:

```json
{
```
Notice:
After the RAM user is authorized to configure tags for specified objects, this user can configure tags only for existing objects.

- Authorize RAM users to manage objects that have specific tags

You can authorize RAM users to manage all objects that have specific tags. For example, you can authorize user A to access all objects that have the tag configuration of allow=yes. An example of the RAM policy is as follows:

```json
{
   "Version": "1",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": "oss:GetObject",
         "Resource": "*",
         "Condition": {
            "StringEquals": {
               "oss:ExistingObjectTag/allow": [ "yes"
            ]
         }
      }
   ]
}
```

9.4 Single-connection bandwidth throttling

OSS allows you to configure bandwidth throttling for upload, download, and copy operations on OSS to ensure sufficient bandwidth for other applications.

Upload and download operations on OSS may consume a large amount of bandwidth. If your client does not have bandwidth throttling features, other applications are affected. To avoid this problem, you can include the `x-oss-traffic`
-limit parameter in your requests to configure bandwidth throttling when you call operations such as PutObject, AppendObject, PostObject, CopyObject, UploadPart, UploadPartCopy, and GetObject.

Scenarios

Single-connection bandwidth throttling applies to the following scenarios:

- You can use the single-connection bandwidth throttling feature for clients such as mobile terminals and Web terminals that do not have advanced throttling features. This can help ensure sufficient bandwidth for other applications.
- When sharing your data on OSS with other parties, you can include the x-oss-traffic-limit parameter in the object URL to limit the download speed. This will ensure sufficient bandwidth for your other business or the access of other users. For example, to share an object file.zip in the root directory of a bucket named atest in the China (Hangzhou) region, you can share the object URL with others for downloads. If you want to set the bandwidth limit to 5 MB/s, the object URL must include the x-oss-traffic-limit=41943040 parameter:

https://atest.oss-cn-hangzhou.aliyuncs.com/file.zip?Expires=1562736800&OSSAccessKeyId=TMP.hW5bxvoEiXZ2BkVvrdBYrfdgcl2RlS8b888DqddZtUButHNBKgmn9ZtvTEBtBFATe4VJpnnmKd48UJomnpS3toHRQXL2FuuSkJgR58RhnD5uRnas6h6ZVHg4tf.tmp&Signature=FhH8m%2FQRDhF%2Bc%2FHFhucUoGRUR8U%3D&x-oss-traffic-limit=41943040

Implementation modes

<table>
<thead>
<tr>
<th>Implementation mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Java SDK</strong></td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td><strong>Python SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Go SDK</strong></td>
<td></td>
</tr>
<tr>
<td><strong>C++ SDK</strong></td>
<td></td>
</tr>
</tbody>
</table>

Precautions

- The x-oss-traffic-limit parameter can be included in headers, request parameters, and form fields for form upload. The parameter must be included in the string to be signed.
- The value of the x-oss-traffic-limit parameter must be a number. It must be expressed in bit/s.
• Valid values of the bandwidth limit is 819200 to 838860800, which is equivalent to 100 KB/s to 100 MB/s.

### Note:
Formula for unit conversion: 1 MB = 1,024 KB =1,048,576 Bytes = 8,388,608 bits

### 9.5 Versioning

#### 9.5.1 Versioning

After versioning is enabled for a bucket, data that is overwritten or deleted is saved as a previous version. Versioning allows you to restore objects in a bucket to any previous point in time after you overwrite or delete the objects.

### Note:
Versioning is now available in India (Mumbai) and Malaysia (Kuala Lumpur) where users are added to the whitelist. This feature will be available in other regions soon.

**Scenarios**

• Restore data that was accidentally deleted

  OSS does not provide functions of recycle bins. After data in OSS is deleted, deleted data cannot be restored. You must use a local or third party backup tool to restore deleted data.

• Restore data that was overwritten

  Documents stored in online storage or online collaborative documents are frequently modified. In online office scenarios, a large number of temporary versions are generated when files are edited. Users need to find versions of a certain point in time.

**Principles**

Versioning applies to all objects instead of specified objects in buckets. After you enable versioning for a bucket, all objects in the bucket are subject to versioning. Each version has a unique version ID.
- Bucket versions can be in any of the following states:
  - Unversioned (default)
  - Versioning-enabled
  - Versioning-suspended

**Note:**
After you have enabled versioning for a bucket, the status of the bucket version can only be suspended. The status cannot be rolled back to unversioned.

- The following users can configure versioning for buckets:
  - Users who have root accounts
  - RAM users or roles who have been granted OSSFullAccess
  - RAM users or roles who have been granted PutBucketVersioning

**Configuration tools**
You can use one of the following methods to configure versioning for a bucket:
- Console
- ossutil

**Note:**
After versioning is enabled for a bucket, fees will be incurred when previous versions are generated for overwritten objects. You can configure lifecycle rules to automatically delete expired versions.

### 9.5.2 Enable versioning
When versioning is enabled for a bucket, OSS assigns a unique ID for each version of all objects in the bucket. The content and ACL of existing objects in the bucket remain unchanged. Versioning prevents unintended overwrites and deletions on your data. You can also use versioning to retrieve a previous version of an object and recover the previous version as the current version at any time.

**Note:**
- By default, versioning is not enabled for a bucket. To enable versioning for a bucket, you must manually configure this feature.
- When versioning is enabled for a bucket, the bucket maintains the current versions and previous versions for its objects.
• When versioning is not enabled for a bucket, OSS sets the object version IDs to null.

OSS allows you to upload objects and list, download, delete, and recover object versions for a versioning-enabled bucket.

Upload objects

When you upload an object to a versioning-enabled bucket, OSS assigns a unique version ID to the object.

Note:

When you perform operations such as PutObject, PostObject, CopyObject, or MultipartUpload on an object, OSS assigns a unique version ID to the object.

When you use the PutObject operation to upload an object with the key of example.jpg, OSS assigns a unique version ID of 111111 to the object. The following figure shows the process.

When you use the PutObject operation to upload the same object with the key of example.jpg, OSS assigns a new unique version ID of 222222 to the object and stores this version as the current version in the bucket. Version 111111 becomes a previous version. The following figure shows the process. When you use the PutObject operation to upload the same object with the key of example.jpg again, OSS assigns a new unique version ID of 333333 to the object and stores this version as the current version in the bucket. Versions 111111 and 222222 become previous versions. The following figure shows the process.

List object versions

You can call the GetBucketVersions(ListObjectVersions) operation to obtain information of all object versions in a versioning-enabled bucket, including delete markers.
Note:

- Unlike GetBucketVersions(ListObjectVersions), the GetBucket (ListObject) operation returns only the current version of the bucket. The current version is not a delete marker.
- Each request can return up to 1,000 versions. To list more versions of an object, you must send multiple requests to retrieve the list of all versions.

For example, if a bucket contains two objects whose names are example.jpg and photo.jpg. The example.jpg object has 900 versions. The photo.jpg object has 500 versions. The first request returns 900 versions of the photo.jpg object and 100 versions of the photo.jpg object. Versions are returned based on the alphabetical order of keys first and then in the order in which these versions are stored.

When you call GetBucketVersions for a versioning-enabled bucket, all object versions in the bucket are returned, including delete markers. When you call GetBucket for a versioning-enabled bucket, only current versions of objects in the bucket are returned and the current versions are not delete markers. The following figure shows the process.

Download object versions

OSS allows you to download an object of the current or a specified version from a versioning-enabled bucket.

By default, the current version of the object is returned if the object version ID is not specified when you use the GetObject request to download an object. The following figure shows that current version (ID: 333333) of the object is returned.
If the current version is a delete marker, the GetObject request returns 404 Not Found.

To download an object of a specified version, specify the object version ID in the GetObject request. The following figure shows that the specified version ID is 222222.

Delete object versions

OSS allows you to specify an object version ID or configure lifecycle rules to permanently delete an object in a versioning-enabled bucket. If you do not specify an object version ID in the DeleteObject request, OSS inserts a delete marker into the bucket, and the delete marker becomes the current version of the object.

Note:
By default, the current version and previous versions of an object are not deleted when you perform the DeleteObject operation on a versioning-enabled bucket.

If you do not specify an object version ID in the DeleteObject request:
• OSS inserts a delete marker into the bucket, and the delete marker becomes the current version of the object. The delete marker has a unique version ID, but does not store any data or have ACL configured. The following figure shows that the current version becomes a delete marker and the version ID is 444444.

• If you specify an object version ID in the DeleteObject request, OSS permanently deletes the object version. The following figure shows that object version 333333 is deleted.

Recover previous versions

When versioning is enabled, all versions of objects in a bucket are preserved. You can recover a specified previous version to make it the current version.

You can use either of the following methods to recover a previous version as the current version:

• Use CopyObject to copy a previous version of an object to the same bucket

The copied object becomes the current version of that object and all object versions are preserved.

In the following figure, a previous version with the version ID of 222222 is copied to the same bucket. OSS assigns a new version ID of 444444 to the object. The new version becomes the current version of the object. Therefore, the object has its previous version of 222222 and its copy 444444 as the current version.
Permanently delete the current version of an object

In the following figure, after you permanently delete the current version of an object by specifying the current version ID of 222222 in the DeleteObject request, the latest previous version of 111111 becomes the current version of the object.

Note:

- We recommend that you use CopyObject to recover a previous version as the current version.
- You can also configure lifecycle rules to delete previous versions of an object.

9.6 FAQ
10 Object lifecycle

10.1 Manage lifecycle rules

You can configure lifecycle rules for OSS buckets or objects by using the `PutBucketLifecycle` interface to automatically delete expired objects and parts or change the storage class of expired objects to IA or Archive, saving storage costs.

Note:
For more information about the `PutBucketLifecycle` interface, see `PutBucketLifecycle`.

A lifecycle rule includes the following information:

- **Matching policy**: Specifies the objects and parts that lifecycle rule applies to.
  - **Matched by prefix**: Indicates that the lifecycle rule applies to objects and parts with the specified prefix. You can create multiple lifecycle rules for objects and parts with different prefixes. The prefix specified in each rule must be different.
  - **Matched by tag**: Indicates that the lifecycle rule applies to objects with the specified tag key and tag value. You can specify multiple tags for a lifecycle rule so that the rule applies to all objects with these tags. Parts cannot be matched to lifecycle rules by tags.

Note:
The object tagging function is in the beta testing phase. You can open a ticket to apply for the trial of this function. For more information about the object tagging function, see `Object tagging`.

- **Matched by prefix and tag**: Indicates that the lifecycle rule applies to objects with the specified prefix and one or more specified tags.
- **Matched to the entire bucket**: Indicates that the lifecycle rule applies to all objects and parts in the bucket. You can create only one lifecycle rule that applies to a bucket.
Object expiration policy: Specifies the expiration time for objects and the operation to be performed on expired objects.

Expiration date: Specifies an expiration date and the operation to be performed on expired objects. An object modified before the specified date expires, and the specified operation is performed on the object.

Note:
Operations that can be performed on an expired object include: convert the storage class of the object to IA, convert the storage class of the object to Archive, and delete the object.

Part expiration policy: Specifies the expiration time for parts and the operation to be performed on expired parts.

- Expiration period: Specifies an expiration period (N days). Parts are deleted N days after it is modified for the last time.
- Expiration date: Specifies an expiration date. All parts modified before the specified date are deleted.

A rule applies to an object if the object name prefix matches the prefix specified in the rule. For example, a bucket includes the following objects:

logs/program.log.1
logs/program.log.2
logs/program.log.3
doc/readme.txt

If the prefix specified in a rule is "logs/", the rule applies to the three objects prefixed with "logs/". If the prefix specified in the rule is "doc/readme.txt", it applies only to the object named doc/readme.txt.

You can also set overdue deletion rules. For example: if the last date of objects that are prefixed with logs/ is 30 days ago, the objects are deleted according to the specified overdue deletion time.

If an object matches an overdue rule, the OSS includes the x-oss-expiration header in the response to the GetObject or HeadObject requests for the object. The header contains two key-value pairs: expiry-date indicates the expiration date of the object, and rule-id indicates the matched rule ID.
Operation method

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSS console</td>
<td>Web application that is easy to use</td>
</tr>
<tr>
<td>Java SDK</td>
<td>Various and complete SDK demos of different languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
<tr>
<td>Node.js SDK</td>
<td></td>
</tr>
<tr>
<td>Ruby SDK</td>
<td></td>
</tr>
</tbody>
</table>

Detail analysis

- **Prefix and tag**
  - The naming conventions for a prefix are the same as those for an object.
  - If the prefix in a rule is empty, the rule applies to all objects in the bucket.
  - Prefixes specified in rules for a bucket must be unique. For example, if two rules are configured for a bucket and the prefixes in the rules are `logs/` and `logs/program` respectively, OSS returns an error.
  - The key of a tag cannot be empty. A tag can include letters, numbers, spaces, and the following symbols:
    + - = . _ : /
  - The prefixes in the matching policies (matched by prefix and tag) of different rules can be overlapped. For example, assume that rule 1 and rule 2 are configured for the same bucket. In rule 1, the specified prefix is `logs/` and the specified tag is "K1=V1". In rule 2, the specified prefix is `logs/program` and the specified tag is "K1=V2". Then, rule 1 applies to objects with the `logs` or `logs/program` prefix and the "K1=V1" tag. Rule 2 applies to objects with the `logs` prefix and the "K1=V1" tag.

- **Effective time**
  - If a rule is set to delete objects on a specified date, the date must be midnight UTC and comply with the ISO8601 format, for example, 2017-01-01T00:00:00.
000Z. In this example, OSS deletes matched objects after midnight on January 1, 2017.

- If a rule is set to delete objects after a specified number of days, OSS sums up the last update time (Last-Modified) and the specified number of days, and then round the sum to the midnight UTC timestamp. For example, if the last update time of an object is 01:00 a.m. on April 12, 2014 and the number of days specified in the matched rule is 3, the expiry time is zero o'clock on April 16, 2017.

- OSS deletes the objects matched the rule at the specified time. Note that objects are usually deleted shortly after the specified time.

- The update time of an unmodified object is typically the time of its creation. If an object undergoes the put operation for multiple times, the last update time is the time of the last Put operation. If an object was copied to itself, the last update time is the time at when the object was last copied.

• Fees

Only successful lifecycle asynchronous request operations are recorded and charged.

• Actions performed when rules conflict

- The prefixes and tags specified in two rules are the same.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Prefix</th>
<th>Tag</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>rule1</td>
<td>abc</td>
<td>a=1</td>
<td>Matched objects are deleted after 20 days.</td>
</tr>
<tr>
<td>Rule</td>
<td>Prefix</td>
<td>Tag</td>
<td>Action</td>
</tr>
<tr>
<td>-------</td>
<td>--------</td>
<td>------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>rule2</td>
<td>abc</td>
<td>a=1</td>
<td>The storage class of matched objects is converted to Archive after 20 days.</td>
</tr>
</tbody>
</table>

Result: Objects with the abc prefix and the "a=1" tag are deleted after 20 days (deletion is performed first). The second rule does not take effect because the matched objects have already been deleted.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Prefix</th>
<th>Tag</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>rule1</td>
<td>abc</td>
<td>a=1</td>
<td>The storage class of matched objects is converted to IA after 365 days.</td>
</tr>
<tr>
<td>rule2</td>
<td>abc</td>
<td>a=1</td>
<td>The storage class of matched objects is converted to Archive before March, 1, 2018.</td>
</tr>
</tbody>
</table>

Result: If objects with the abc prefix and the "a=1" tag match the two rules at the same time, the storage of the objects is converted to Archive. If the objects only match one of the rules, the action specified in the matched rule is performed.

- The tags specified in two rules are the same, and the prefixes specified in the rules are overlapped.
<table>
<thead>
<tr>
<th>Rule</th>
<th>Prefix</th>
<th>Tag</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>rule2</td>
<td>abc</td>
<td>a=1</td>
<td>Matched objects are deleted after 120 days.</td>
</tr>
</tbody>
</table>

Result: The storage class of all objects with the "a=1" tag is converted to IA after 20 days. Objects with the abc prefix and the "a=1" tag are deleted after 120 days.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Prefix</th>
<th>Tag</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>rule1</td>
<td>-</td>
<td>a=1</td>
<td>The storage class of matched objects is converted to Archive after 10 days.</td>
</tr>
<tr>
<td>rule2</td>
<td>abc</td>
<td>a=1</td>
<td>The storage class of matched objects is converted to IA after 20 days.</td>
</tr>
</tbody>
</table>

Result: The storage class of all objects with the "a=1" tag is converted to Archive. The second rule does not take effect on objects with the abc prefix and the "a=1" tag because the storage class of Archive objects cannot be converted to IA.

### 10.2 Lifecycle rule configuration examples

This topic provides examples of lifecycle rule configuration.

You can set lifecycle rules for objects in a bucket through OSS APIs. A lifecycle rule is in XML format, as shown in the following example:

```xml
<LifecycleConfiguration>
  <Rule>
    <ID>delete logs after 10 days</ID>
    <Prefix>logs/</Prefix>
    <Status>Enabled</Status>
    <Expiration>
      <Days>10</Days>
    </Expiration>
  </Rule>
  <Rule>
    <ID>delete doc</ID>
  </Rule>
</LifecycleConfiguration>
```
In the preceding example, the following three lifecycle rules are set:

- The first rule indicates that objects that are prefixed with logs/ and were modified 10 days ago are deleted.
- The second rule indicates that objects that are prefixed with doc/ and were modified before December 31, 2014 are deleted. However, the rule does not take effect because it is in Disabled status.
- The third rule indicates that the storage class of objects that are tagged with "xx=1" and were modified 60 days ago is converted to Archive.

You must configure the following elements when setting lifecycle rules:

- <ID>: Indicates the unique identifier for a rule.
- <Status>: Indicate the status of the lifecycle rule with two values: Enabled or Disabled. Only rules with the Enabled status is applied.
- <Prefix>: Indicates that the rule applies only to objects with the specified prefix.
- <Expiration>: Indicates operation to be performed on expired objects. The sub-element <CreatedBeforeDate> and <Days> indicates the absolute and relative expiration time respectively.
  - <CreatedBeforeDate>: Specifies an expiration date and the operation to be performed on expired objects. An object modified before the specified date expires, and the specified operation is performed on the object.
  - <Days> Specifies an expiration period (N days) and the operation to be performed on expired objects. An object expires N days after it is modified for the last time, and the specified operation is performed on the object.
10.3 FAQ

This topic answers the problems that may occur when you use lifecycle rules to manage objects.

Is there a minimum storage period for a lifecycle rule if the rule is set to convert storage class or delete an expired object?

If a lifecycle rule is set to convert the storage class (from Standard to IA or Archive or from IA to Archive) of objects, no minimum storage period is required for the rule. However, a lifecycle rule set to delete expired objects requires a minimum storage period. The minimum storage period is 30 days for objects of the IA storage class and 60 days for objects of the Archive storage class.

The minimum storage period for a lifecycle rule set to delete an expired object indicates the time period from the time when the object is created until the time when the object is deleted. If the object is modified (such as by CopyObject or AppendObject operations) during this period, the minimum storage period is calculated from the last modified date.

Example 1: The storage class of an object is converted from IA to Archive 10 days after it is created. The creation time of the object does not change. The converted object must be stored for at least 50 days.

Example 2: If the storage class of an object is converted from IA to Archive through a CopyObject operation 10 days after it is created, fees of 20 days are incurred for
the deletion. The creation time of the object updates. The converted object must be stored for at least 60 days.

Billing logic for requests

The conversion of storage class or the deletion of expired objects performed according to lifecycle rules generate requests. The requests are charged by OSS. For example:

- If the storage class of 1000 objects are converted from Standard to Archive according to a lifecycle rule, 1000 POST requests are generated.
- If 1000 expired objects are deleted according to a lifecycle rule, 1000 Delete requests are generated.

For more information, see Billing methods.

Are the conversion of storage class or the deletion of expired objects performed according to lifecycle rules recorded?

The conversion of storage class or the deletion of expired objects performed according to lifecycle rules are recorded in logs. Fields in the logs are described as follows:
• Operation
  - CommitTransition: Indicates that the rule is set to convert the storage class of object.
  - ExpireObject: Indicates that the rule is set to delete expired objects.

• Sync Request

  lifecycle: Indicates the operation to be performed by the lifecycle rule.
11 Signature

11.1 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 account 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.

11.2 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:
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.
Object Storage Service Developer Guide / 11 Signature

- 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\nx-oss-meta-b:b\nx-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 OtxrzxIsfpFjA7SwPzILwy8Bw21TLhqubboDYROV.
The signature calculation method is as follows:

**Python sample code:**

```python
import base64
import hmac
import sha
h = hmac.new("OtxrzxIsfpFjA7SwPzILwy8Bw21TLhqubboDYROV",
"PUT
ODBG0ERFMDMzQTczRUY3NUE3NzA5QzdFNUYzMDQxNEM=\n\ntext/html
Thu, 17 Nov 2005
18:49:58 GMT
x-oss-magic:abracadabra
x-oss-meta-author:foo@bar.com
/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
Authorization:OSS 44CF9590006BF252F707:26NBxoKdsyly4EDv6inkoDft/yA=
Content-Md5: eB5eJF1ptWaXm4bijdPyxw==
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
```

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:

date = 2DIGIT SP month SP 4DIGIT; day month year (for example, 02 Jun 1982). In the above 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
<?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 73 74 3f 61 63 6c</StringToSignBytes>
  <RequestId>1E446260FF9B10C2</RequestId>
  <HostId>oss-cn-hangzhou.aliyuncs.com</HostId>
  <SignatureProvided>y5H7yzPsA/tP4+0tH1HHvPEwUv8=</SignatureProvided>
  <StringToSign>GET
  Wed, 11 May 2011 07:59:25 GMT
  /oss-example? acl</StringToSign>
  <OSSAccessKeyId>AKIAIVAKMSMOY7VOMRWQ</OSSAccessKeyId>
</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:
```python
>>> import base64, hashlib
>>> hash = hashlib.md5()
>>> hash.update("0123456789")
>>> base64.b64encode(hash.digest())
'eB5eJF1ptWaXm4bijiSPyxw=='
```
Note:
The correct code is: hash.digest(), used to calculate a 128-bit binary array
```python
>>> 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.
```python
>>> hash.hexdigest()
'781e5e245d69b566979b86e28d23f2c7'
```
Result of encoding the incorrect MD5 value with Base64:
```python
>>> base64.b64encode(hash.hexdigest())
'NzgxZTVlmjQ1ZDY5YjU2Njk30WI4NnUyOGQyM2YyYzc='
```

11.3 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:
```python
import base64
import hmac
import sha
import urllib
h = hmac.new("OtxrzxIsfFjA7SwPzILwy8Bw21TLhquhboDYROV",
             "GET\n\n\n1141889120\n/oss-example/oss-api.pdf",
             sha)
urllib.quote (base64.encodestring(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.

<table>
<thead>
<tr>
<th>SDK</th>
<th>URL signature method</th>
<th>Implementation file</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java SDK</td>
<td>OSSClient.generatePresignedUrl</td>
<td>OSSClient.java</td>
</tr>
<tr>
<td>Python SDK</td>
<td>Bucket.sign_url</td>
<td>api.py</td>
</tr>
<tr>
<td>Net SDK</td>
<td>OssClient.GeneratePresignedUri</td>
<td>OssClient.cs</td>
</tr>
<tr>
<td>PHP SDK</td>
<td>OssClient.signUrl</td>
<td>OssClient.php</td>
</tr>
<tr>
<td>JavaScript SDK</td>
<td>signatureUrl</td>
<td>object.js</td>
</tr>
<tr>
<td>C SDK</td>
<td>oss_gen_signed_url</td>
<td>oss_object.c</td>
</tr>
</tbody>
</table>

Implementation

URL signature example:

```text
http://oss-example.oss-cn-hangzhou.aliyuncs.com/oss-api.pdf?OSSAccessKeyId=nz2pc56s936**9l&Expires=1141889120&Signature=vjbyPxybdZaNmGa%2ByT272YEAiv4%3D
```

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

- 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.

- OSSAccessKeyId 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.

```python
Signature = urlencode(base64(hmac-sha1(AccessKeySecret, VERB + "\n" + CONTENT-MD5 + "\n" + CONTENT-TYPE + "\n"))
```
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 UrleEncode 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?
OSSAccessKeyId=nz2pc56s936**9l&Expires=1141889120&Signature=
vjbyPxybdZaNmGa%2ByT272YEAiv4%3D&security-token=SecurityToken
```

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.)
12 Identity authentication

12.1 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?

RM 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 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’s 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 osscmd, a tool in the OSS PythonSDK that allows you to directly work on OSS through the command line. osscmd can be obtained from PythonSDK.

Typical osscmd usage:

```
Download files
./osscmd 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 corresponding to your own account
Upload files
./osscmd put LOCALFILE oss://BUCKET/OBJECT --host=Endpoint -i AccessKeyId -k AccessKeySecret
The meaning of each field is the same as for the download example
```

12.2 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 that you authorize. 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 #unique_290. To grant OSS access permissions to users by creating RAM policies, see RAM Policy.

### 12.3 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 AccessKey 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.

![Create User](image)

- **Add User**

<table>
<thead>
<tr>
<th>Login Name</th>
<th>Display Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAMTest</td>
<td><a href="mailto:RamTest@123.onaliyun.com">RamTest@123.onaliyun.com</a></td>
</tr>
</tbody>
</table>

- **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.

![Add Permissions](image)

- **RAMTest@123.onaliyun.com**

- **Basic Information**
  - Username: RAMTest
  - Display Name: RamTest
  - Note: Mobile Phone Number
  - Created: Dec 21, 2018 14:29:30

- **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.
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 andGetObject, to a RAM user named ram-test, add the following script in the Policy Document.

   ```json
   {
       "Version": "1",
       "Statement": [ {
           "Effect": "Allow",
           "Action": [ "oss:ListObjects", "oss:GetObject" ],
           "Resource": [ "acs:oss:*:*:ram-test", "acs:oss:*:*:ram-test/*" ]
       } ]
   }
   ```
### Create Custom Policy

<table>
<thead>
<tr>
<th>Policy Name</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Configuration Mode**

- Visualized
- Solar

**Policy Document**

```
{ "Version": 1,  
  "Statement": [ 
    { 
      "Sid": "DenyGetObject", 
      "Effect": "Deny", 
      "Action": [ "s3:GetObject" ], 
      "Resource": [ 
        "arn:aws:s3:::my-bucket/my-object" 
      ] 
    } 
  ] 
}
```

**Import an existing system policy**

- Click button
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](image)

   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.
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.

```java
public class StsServiceSample {
    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 = 
            "    "Version": "1", 
            "Statement": [
                {
                    "Action": [
                        "oss:*"
                    ], 
                    "Resource": [
                        "acs:oss:*:*:*" 
                    ], 
                    "Effect": "Allow"
                }
            ]
        
        try {
            // Adds an endpoint. (The STS endpoint is directly used
            DefaultProfile.addEndpoint("", "", "Sts", endpoint);
            // Constructs a default profile. (The parameter is left
            IClientProfile profile = DefaultProfile.getProfile("", accessKeyId, accessKeySecret);
            // Uses the constructed profile to construct a client.
            DefaultAcsClient client = new DefaultAcsClient(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.getErrCode());
    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.

```java
// 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
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
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();
```
13 Access and control

13.1 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.

13.2 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.

<table>
<thead>
<tr>
<th>ACL</th>
<th>Description</th>
<th>Access control</th>
</tr>
</thead>
<tbody>
<tr>
<td>public-read-write</td>
<td>The public-read-write permission.</td>
<td>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.</td>
</tr>
<tr>
<td>public-read</td>
<td>The public-read permission.</td>
<td>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.</td>
</tr>
<tr>
<td>private</td>
<td>The private permission.</td>
<td>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.</td>
</tr>
</tbody>
</table>
• Operating methods

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

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.

<table>
<thead>
<tr>
<th>ACL</th>
<th>Description</th>
<th>Access control</th>
</tr>
</thead>
<tbody>
<tr>
<td>public-read-write</td>
<td>The public-read-write permission.</td>
<td>All users can read data from and write data to the object.</td>
</tr>
<tr>
<td>public-read</td>
<td>The public-read permission.</td>
<td>The object owner can read data from and write data to the object. Other users can only read data from the object.</td>
</tr>
</tbody>
</table>
### Access and control

<table>
<thead>
<tr>
<th>ACL</th>
<th>Description</th>
<th>Access control</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>The private permission.</td>
<td>The object owner can read data from and write data to the object. Without authorization, other users have no access to the object.</td>
</tr>
<tr>
<td>default</td>
<td>The default permission.</td>
<td>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.</td>
</tr>
</tbody>
</table>

**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

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

Reference

For more information about how to allow only specified users to access your objects, see the following topics:
13.3 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.

13.4 Access control based on RAM Policy

13.4.1 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.

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

<table>
<thead>
<tr>
<th>Operations on OSS</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>List all created buckets.</td>
<td>Success</td>
</tr>
<tr>
<td>Upload an object without a prefix, such as text.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>Download an object without a prefix, such as text.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>Upload an object with a prefix, such as user1/test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>Download an object with a prefix, such as user1/test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>List objects without prefixes, such as test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>List objects with prefixes, such as user1/test.txt.</td>
<td>Success</td>
</tr>
</tbody>
</table>

- **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.

  ```json
  {
    "Statement": [
    ```
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 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.

```json
{
  "Action": [
    "oss:GetObject",
    "oss:ListObjects"
  ],
  "Effect": "Allow",
  "Resource": ["acs:oss:***:app-base-oss/*", "acs:oss:***:app-base-oss/user1/*",
    "acs:oss:***:app-base-oss"
  ],
  "Version": "1"
}
```
## Operations on OSS

<table>
<thead>
<tr>
<th>Operations on OSS</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>List all created buckets.</td>
<td>Failed</td>
</tr>
<tr>
<td>Upload an object without a prefix, such as text.txt.</td>
<td>Failed</td>
</tr>
<tr>
<td>Download an object without a prefix, such as text.txt.</td>
<td>Failed</td>
</tr>
<tr>
<td>Upload an object with a prefix, such as user1/test.txt.</td>
<td>Failed</td>
</tr>
<tr>
<td>Download an object with a prefix, such as user1/test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>List objects without prefixes, such as test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>List objects with prefixes, such as user1/test.txt.</td>
<td>Success</td>
</tr>
</tbody>
</table>

### 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"
    }
  ]
}
```

<table>
<thead>
<tr>
<th>Operations on OSS</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>List all created buckets.</td>
<td>Failed</td>
</tr>
<tr>
<td>Upload an object without a prefix, such as text.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>Download an object without a prefix, such as text.txt.</td>
<td>Failed</td>
</tr>
<tr>
<td>Upload an object with a prefix, such as user1/test.txt.</td>
<td>Success</td>
</tr>
</tbody>
</table>
### Operations on OSS

<table>
<thead>
<tr>
<th>Operations on OSS</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Download an object with a prefix, such as user1/test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>List objects without prefixes, such as test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>List objects with prefixes, such as user1/test.txt.</td>
<td>Success</td>
</tr>
</tbody>
</table>

- **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.

  ```json
  {
    "Statement": [
      {
        "Action": [
          "oss:PutObject"
        ],
        "Effect": "Allow",
        "Resource": ["acs:oss::*:app-base-oss/user1/*", "acs:oss::*:app-base-oss"]
      }
    ],
    "Version": "1"
  }
  ```

### Operations on OSS

<table>
<thead>
<tr>
<th>Operations on OSS</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>List all created buckets.</td>
<td>Failed</td>
</tr>
<tr>
<td>Upload an object without a prefix, such as text.txt.</td>
<td>Failed</td>
</tr>
<tr>
<td>Download an object without a prefix, such as test.txt.</td>
<td>Failed</td>
</tr>
<tr>
<td>Upload an object with a prefix, such as user1/test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>Download an object with a prefix, such as user1/test.txt.</td>
<td>Failed</td>
</tr>
<tr>
<td>List objects without prefixes, such as test.txt.</td>
<td>Failed</td>
</tr>
<tr>
<td>List objects with prefixes, such as user1/test.txt.</td>
<td>Failed</td>
</tr>
</tbody>
</table>
• 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:AbortMultipartUpload",
         "oss:ListObjects"
      ],
      "Effect": "Allow",
      "Resource": ["acs:oss:*:*:app-base-oss/*", "acs:oss:*:*:app-base-oss"]
   }
   ],
   "Version": "1"
}
```

<table>
<thead>
<tr>
<th>Operations on OSS</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>List all created buckets.</td>
<td>Failed</td>
</tr>
<tr>
<td>Upload an object without a prefix, such as text.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>Download an object without a prefix, such as text.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>Upload an object with a prefix, such as user1/test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>Download an object with a prefix, such as user1/test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>List objects without prefixes, such as test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>List objects with prefixes, such as user1/test.txt.</td>
<td>Success</td>
</tr>
</tbody>
</table>

• 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.

```json
{
    "Statement": [
        {
            "Action": [
                "oss:GetObject",
                "oss:PutObject",
                "oss:DeleteObject",
                "oss:ListParts",
                "oss:AbortMultipartUpload",
                "oss:ListObjects"
            ],
            "Effect": "Allow",
            "Resource": [
                "acs:oss:*:*:app-base-oss/user1/*",
                "acs:oss:*:*:app-base-oss"
            ]
        },
        "Version": "1"
    }
}
```

<table>
<thead>
<tr>
<th>Operations on OSS</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>List all created buckets.</td>
<td>Failed</td>
</tr>
<tr>
<td>Upload an object without a prefix, such as text.txt.</td>
<td>Failed</td>
</tr>
<tr>
<td>Download an object without a prefix, such as text.txt.</td>
<td>Failed</td>
</tr>
<tr>
<td>Upload an object with a prefix, such as user1/test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>Download an object with a prefix, such as user1/test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>List objects without prefixes, such as test.txt.</td>
<td>Success</td>
</tr>
<tr>
<td>List objects with prefixes, such as user1/test.txt.</td>
<td>Success</td>
</tr>
</tbody>
</table>

Complex policy examples

```json
{
    "Version": "1",
    "Statement": [
        {
            "Action": [
                "oss:GetBucketAcl",
                "oss:ListObjects"
            ],
            "Resource": [
                "acs:oss:*:*:mybucket"
            ]
        },
        "Version": "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

<table>
<thead>
<tr>
<th>API</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetService (ListBuckets)</td>
<td>oss:ListBuckets</td>
</tr>
</tbody>
</table>

- Bucket-level actions

<table>
<thead>
<tr>
<th>API</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>PutBucket</td>
<td>oss:PutBucket</td>
</tr>
<tr>
<td>GetBucket (ListObjects)</td>
<td>oss:ListObjects</td>
</tr>
</tbody>
</table>
### API Actions

<table>
<thead>
<tr>
<th>API</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>PutBucketAcl</td>
<td>oss:PutBucketAcl</td>
</tr>
<tr>
<td>DeleteBucket</td>
<td>oss:DeleteBucket</td>
</tr>
<tr>
<td>GetBucketLocation</td>
<td>oss:GetBucketLocation</td>
</tr>
<tr>
<td>GetBucketAcl</td>
<td>oss:GetBucketAcl</td>
</tr>
<tr>
<td>GetBucketLogging</td>
<td>oss:GetBucketLogging</td>
</tr>
<tr>
<td>PutBucketLogging</td>
<td>oss:GetBucketLogging</td>
</tr>
<tr>
<td>DeleteBucketLogging</td>
<td>oss:GetBucketLogging</td>
</tr>
<tr>
<td>GetBucketWebsite</td>
<td>oss:GetBucketWebsite</td>
</tr>
<tr>
<td>PutBucketWebsite</td>
<td>oss:GetBucketWebsite</td>
</tr>
<tr>
<td>DeleteBucketWebsite</td>
<td>oss:GetBucketWebsite</td>
</tr>
<tr>
<td>GetBucketReferer</td>
<td>oss:GetBucketReferer</td>
</tr>
<tr>
<td>PutBucketReferer</td>
<td>oss:GetBucketReferer</td>
</tr>
<tr>
<td>GetBucketLifecycle</td>
<td>oss:GetBucketLifecycle</td>
</tr>
<tr>
<td>PutBucketLifecycle</td>
<td>oss:GetBucketLifecycle</td>
</tr>
<tr>
<td>DeleteBucketLifecycle</td>
<td>oss:GetBucketLifecycle</td>
</tr>
<tr>
<td>ListMultipartUploads</td>
<td>oss:ListMultipartUploads</td>
</tr>
<tr>
<td>PutBucketCors</td>
<td>oss:GetBucketCors</td>
</tr>
<tr>
<td>GetBucketCors</td>
<td>oss:GetBucketCors</td>
</tr>
<tr>
<td>DeleteBucketCors</td>
<td>oss:GetBucketCors</td>
</tr>
<tr>
<td>PutBucketReplication</td>
<td>oss:GetBucketReplication</td>
</tr>
<tr>
<td>GetBucketReplication</td>
<td>oss:GetBucketReplication</td>
</tr>
<tr>
<td>DeleteBucketReplication</td>
<td>oss:GetBucketReplication</td>
</tr>
<tr>
<td>GetBucketReplicationLocation</td>
<td>oss:GetBucketReplicationLocation</td>
</tr>
<tr>
<td>GetBucketReplicationProgress</td>
<td>oss:GetBucketReplicationProgress</td>
</tr>
</tbody>
</table>

### Actions on objects

<table>
<thead>
<tr>
<th>API</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetObject</td>
<td>oss:GetObject</td>
</tr>
<tr>
<td>HeadObject</td>
<td>oss:GetObject</td>
</tr>
<tr>
<td>PutObject</td>
<td>oss:GetObject</td>
</tr>
</tbody>
</table>
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.

```json
{
  "Version": "1",
  "Statement": [
```
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**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Function</th>
<th>Valid value</th>
</tr>
</thead>
<tbody>
<tr>
<td>acs:SourceIp</td>
<td>Specifies the source IP address or IP range.</td>
<td>IP address or IP range, wildcards (*) supported</td>
</tr>
<tr>
<td>acs:UserAgent</td>
<td>Specifies the http useragent header.</td>
<td>String</td>
</tr>
<tr>
<td>acs:CurrentTime</td>
<td>Specifies a valid access time.</td>
<td>Time in the ISO8601 format</td>
</tr>
<tr>
<td>acs:SecureTransport</td>
<td>Indicates whether the HTTPS protocol is used.</td>
<td>&quot;true&quot; or &quot;false&quot;</td>
</tr>
<tr>
<td>oss:Prefix</td>
<td>Indicates the prefix used when the ListObjects action is performed.</td>
<td>Valid object name</td>
</tr>
</tbody>
</table>

**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.

13.4.2 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:
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.
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
```
OSS returns a response that includes the `ListBucketResult` element:

```
HTTP/1.1 200 OK
x-oss-request-id: 534B371674E88A4D8906008B
Date: Fri, 24 Feb 2012 08:43:27 GMT
Content-Type: application/xml
Content-Length: 712
Connection: keep-alive
Server: AliyunOSS
<? xml version="1.0" encoding="UTF-8" ?>
<ListBucketResult xmlns="http://doc.oss-cn-hangzhou.aliyuncs.com;">
 <Name>example-company</Name>
 <Prefix/>
 <Marker/>
 <MaxKeys>100</MaxKeys>
 <Delimiter>/</Delimiter>
 <IsTruncated>false</IsTruncated>
 <Contents>
   <Key>oss-dg.pdf</Key>
   ...
 </Contents>
 <CommonPrefixes>
   <Prefix>Development</Prefix>
 </CommonPrefixes>
 <CommonPrefixes>
   <Prefix>Marketing</Prefix>
 </CommonPrefixes>
 <CommonPrefixes>
   <Prefix>Private</Prefix>
 </CommonPrefixes>
</ListBucketResult>
```

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 `CommonPrefixes/` element for each of the prefix values `Development/`, `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:

```plaintext
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 qn6qrrqxo2oawuk53otfjbyc:DNrnx7xHk3sgysx7I8U9I9IY1vY=
```

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

```plaintext
HTTP/1.1 200 OK
x-oss-request-id: 534B371674E88A4D8906008B
Date: Fri, 24 Feb 2012 08:43:27 GMT
Content-Type: application/xml
```
The console interprets this result and displays the object keys as follows:

![Console Interface]

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.
   
   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:ListBuckets` 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:ListObjects` 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.

```json
{
   "Version": "1",
   "Statement": [
   {
      "Effect": "Allow",
      "Action": [
      "oss:ListBuckets",
      "oss:GetBucketStat",
      "oss:GetBucketInfo",
      "oss:GetBucketAcl"
      ],
      "Resource": [
      "acs:oss:*:*:*"
      ]
   }
   ]
}
```
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.

![Image of OSS console and message box]

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 `AllowGroupToSeeBucketListInConsole` 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, created a new one, and then attach the policy to the Staff group again.

   ```json
   {
     "Version": "1",
     "Statement": [
       {
         "Effect": "Allow",
         "Action": [
           "oss:ListObjects"
         ]
       }
     ]
   }
   ```
"oss:ListBuckets",
"oss:GetBucketStat",
"oss:GetBucketInfo",
"oss:GetBucketAcl"
],
"Resource": [  "acs:oss:*:*:*"
],
"Condition": {}},
{
"Effect": "Allow",
"Action": [  "oss:ListObjects"
],
"Resource": [  "acs:oss:*:*:example-company"
],
"Condition": {
"StringLike": {
"oss:Prefix": [  ""
],
"oss:Delimiter": [  "/"  ]
}
}
]
]
)

Note:

• To list bucket content, users need permission to call the `oss:ListObjects` 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:GetBucketAcl` 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.

![example-company]

   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.

```json
{
    "Version": "1",
    "Statement": [{
        "Sid": "AnneListFolderContent",
        "Effect": "Allow",
        "Action": "oss:ListObjects",
        "Resource": "arn:aliyun:oss::example-company:development/",
        "Condition": {
            "StringLike": {
                "aws:PrincipalID": [
                    "acs:aliyun.com:*:user/Anne"
                ]
            }
        }
    }]
}
```
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 **Development/** folder. The console lists the objects in the folder.

**Step 5.2 Grant RAM User Anne permissions to get and put objects in the Development folder**

For Anne to get and put objects in the **Development** 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 `Development/*` in the request.

1. **Log on to the **RAM console** with your Alibaba Cloud account credentials.**

2. **Replace the policy** `AllowListBucketsIfSpecificPrefixIsIncluded` **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.

```json
{
    "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": {}
        }
    ]
}
```

**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:GetObjectAcl` 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 `Development` 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/*"
          ]
        }
      }
    }
  ]
}
```
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/*`).

```json
{
  "Effect": "Deny",
  "Action": [
    "oss:*"
  ],
  "Resource": [
    "acs:oss:*:*:example-company/Private/*"
  ],
  "Condition": {}
}
```

- You also deny permission for the `ListObjects` 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.

```json
{
  "Effect": "Deny",
  "Action": [
    "oss:ListObjects"
  ],
  "Resource": [
    "acs:oss:*:*:*"
  ],
  "Condition": {
    "StringLike": {
      "oss:Prefix": [
        "Private/"
      ]
    }
  }
}
```

- Replace the Staff group policy `AllowGroupToSeeBucketListInConsole` 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 `AllowGroupToSeeBucketListInConsole` 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.

```json
{
  "Version": "1",
  "Statement": [
    
  ]
}
```
"Effect": "Allow",
  "Action": [
    "oss:ListBuckets",
    "oss:GetBucketStat",
    "oss:GetBucketInfo",
    "oss:GetBucketAcl"
  ],
  "Resource": [
    "acs:oss::*::*::*"
  ],
  "Condition": {}],
{
  "Effect": "Allow",
  "Action": [
    "oss:ListObjects"
  ],
  "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:ListObjects"
  ],
  "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.

13.5 Cross-account authorization

13.5.1 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.

13.5.2 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 Files > 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

Tutorial: Authorize a RAM user under another Alibaba Cloud account by creating a RAM role
14 Manage logs

14.1 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

<table>
<thead>
<tr>
<th>Operating method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>Web application, which is intuitive and easy to use</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
<tr>
<td>Node.js SDK</td>
<td></td>
</tr>
<tr>
<td>Ruby SDK</td>
<td></td>
</tr>
</tbody>
</table>
Naming rules for objects that store access logs

\[
\text{<TargetPrefix><SourceBucket>YYYY-mm-DD-HH-MM-SS-UniqueString}
\]

In the naming rules:

- **TargetPrefix**: 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remote IP</td>
<td>119.xxx.xx.11</td>
<td>The IP address from which the request is initiated. (The proxy or user firewall may block this field.)</td>
</tr>
<tr>
<td>Reserved</td>
<td>None</td>
<td>The reserved field.</td>
</tr>
<tr>
<td>Reserved</td>
<td>None</td>
<td>The reserved field.</td>
</tr>
<tr>
<td>Time</td>
<td>[02/May/2012:00:00:04 + 0800]</td>
<td>The time OSS receives the request.</td>
</tr>
<tr>
<td>Name</td>
<td>Example</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Request-URI</td>
<td>&quot;GET /aliyun-logo.png HTTP/1.1&quot;</td>
<td>The URI of the user request, including query-string.</td>
</tr>
<tr>
<td>HTTP Status</td>
<td>200</td>
<td>The HTTP status code returned by OSS.</td>
</tr>
<tr>
<td>SentBytes</td>
<td>5576</td>
<td>The amount of traffic downloaded by the user from OSS.</td>
</tr>
<tr>
<td>RequestTime (ms)</td>
<td>71</td>
<td>The duration used to complete the request, in milliseconds.</td>
</tr>
<tr>
<td>User-Agent</td>
<td>curl/7.15.5</td>
<td>The User-Agent field in the HTTP header.</td>
</tr>
<tr>
<td>HostName</td>
<td>oss-example.oss-cn-hangzhou.aliyuncs.com</td>
<td>The domain to be accessed.</td>
</tr>
<tr>
<td>Request ID</td>
<td>505B016950xxxxxx032593A4</td>
<td>The UUID used to uniquely identify the request.</td>
</tr>
<tr>
<td>LoggingFlag</td>
<td>true</td>
<td>Indicates whether the access logging feature is enabled.</td>
</tr>
<tr>
<td>Requester Aliyun ID</td>
<td>16571xxxxxx83691</td>
<td>The RAM user ID, which is a hyphen (-) for anonymous access.</td>
</tr>
<tr>
<td>Operation</td>
<td>GetObject</td>
<td>The request type.</td>
</tr>
<tr>
<td>Bucket</td>
<td>oss-example</td>
<td>The name of the bucket to be accessed.</td>
</tr>
<tr>
<td>Key</td>
<td>/aliyun-logo.png</td>
<td>The name (key) of the object that the user requests.</td>
</tr>
<tr>
<td>ObjectSize</td>
<td>5576</td>
<td>The object size.</td>
</tr>
<tr>
<td>Server Cost Time (ms)</td>
<td>17</td>
<td>The duration for the OSS server to process this request, in milliseconds.</td>
</tr>
<tr>
<td>Name</td>
<td>Example</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Error Code</td>
<td>NoSuchBucket</td>
<td>The error code returned by OSS.</td>
</tr>
<tr>
<td>Request Length</td>
<td>302</td>
<td>The length of the user request, in bytes.</td>
</tr>
<tr>
<td>UserID</td>
<td>16571xxxxxx83691</td>
<td>The ID of the bucket owner.</td>
</tr>
<tr>
<td>Delta DataSize</td>
<td>280</td>
<td>The bucket size variation, which is a hyphen (−) if the bucket size does not change.</td>
</tr>
<tr>
<td>Sync Request</td>
<td>None</td>
<td>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.</td>
</tr>
<tr>
<td>Reserved</td>
<td>None</td>
<td>The reserved field.</td>
</tr>
</tbody>
</table>

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.

### 14.2 Real-time log query

When you access OSS, a large number of access logs are generated. By combining OSS and *Log Service (LOG)*, the real-time log query feature allows you to query OSS access logs in the OSS console and implement operation audit, access statistics collection, exception backtracking, and troubleshooting accordingly. With this feature, you can improve your work efficiency and make decisions based on data.

**Comparison between real-time log query and access logging**

- **Real-time log query:**
  - Pushes logs to Log Service instances within 3 minutes and allows you to view real-time logs in the OSS console.
  - Provides the log analysis service and typical analysis reports so that you can easily query data.
  - Allows you to query and analyze raw logs in real time and filter logs by bucket, object name, API operation, and time.
• Access logging:
  - Allows you to enable the logging feature for a bucket. Then, 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.
  - Allows you to use Alibaba Cloud Data Lake Analytics or build a Spark cluster to analyze access logs.
  - Allows you to set lifecycle management rules for the specified bucket to convert the storage class of log objects to Archive for long-term archiving.

Configuration method

**Console:** *Enable real-time log query*

Query methods

The real-time log query feature provides the following query methods:

• Real-time raw log query

  You can specify the period and query statement for raw logs to easily perform the following operations:
  - Analyze the distribution of a field, such as an API, within a period.
  - Filter records to be analyzed by field for further queries. For example, filter the object deletion operations for the last day by bucket, object name, or API name, and query the deletion time and access IP address.
  - Collect statistics on OSS access records, for example, the page view (PV), unique visitor (UV), or maximum latency of a bucket within a period.

• Log report query

The real-time log query feature provides four immediately available reports:

- Access Center: displays the overall operating status, including the PV, UV, traffic, and distribution of Internet access.
- Audit Center: displays statistics for object operations, including read, write, and delete operations on objects.
- Operation Center: displays statistics for access logs, including the number of requests and distribution of failed operations.
- Performance Center: displays statistics for performance, including the performance of download or upload through the Internet, the performance of
transmission through different networks or with different object sizes, and the list of differences between stored and downloaded objects.

- Query through the Log Service console

You can view OSS access logs in the Log Service console.

Billing method

The real-time log query feature of OSS allows you to query logs for the past seven days free of charge. If the log storage time that you set is longer than seven days, Log Service separately charges the excess days. Extra fees are charged when you read data from or write data to Log Service through the Internet.

For more information about the billing standards, see Billing method of Log Service.
15 Data encryption

15.1 Server-side encryption

OSS supports server-side encryption for uploaded data. When you upload data, OSS encrypts the data and stores the encrypted data. When you download data, OSS automatically decrypts the data and returns the original data to the user. The returned HTTP request header declares that the data has been encrypted on the server.

⚠️ Notice:
Server-side encryption cannot automatically encrypt data retrieved by the mirroring-based back-to-origin feature.

Scenarios

OSS protects static data through server-side encryption. This method is suited for scenarios that require additional security or compliance for object storage. Examples include the storage of deep learning samples and online collaborative documents. You can choose either of the following methods to implement server-side encryption depending on how you choose to manage the encryption keys:

- Implement server-side encryption with CMKs stored in KMS (SSE-KMS)

  When uploading an object, you can use a specified CMK ID or the default CMK stored in KMS to encrypt and decrypt large amounts of data. This method is cost-effective because you do not need to send user data to the KMS server through networks for encryption and decryption.

  ⚠️ Notice:
You will be charged for making API calls when you use CMKs to encrypt or decrypt data.

- Implement server-side encryption with OSS-managed keys (SSE-OSS)

  This encryption method is an attribute of objects. In this method, OSS server-side encryption uses AES-256 to encrypt objects with different data keys. Master keys used to encrypt data keys are rotated regularly. This method is suited to encrypt and decrypt multiple objects at a time.
Notice:
Only one server-side encryption method can be used for an object at a time.

Implementation modes

<table>
<thead>
<tr>
<th>Implementation mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Console</td>
<td>A user-friendly and intuitive Web application</td>
</tr>
<tr>
<td>ossutil</td>
<td>A high-performance command-line tool</td>
</tr>
<tr>
<td></td>
<td>• Configure the default server-side encryption method for a bucket</td>
</tr>
<tr>
<td></td>
<td>• Upload an object and specify a server-side encryption method for the object</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos for various programming languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
</tbody>
</table>

Principles

- Server-side encryption through CMKs stored in KMS

  KMS is a secure and easy-to-use management service provided by Alibaba Cloud. KMS ensures the privacy, integrity, and availability of your keys at minimal cost and allows you to securely and conveniently use keys. You can develop encryption and decryption solutions that best suit your needs. You can view and manage keys in the KMS console.

  KMS encrypts data based on AES-256 and stores and manages CMKs used to encrypt data keys. KMS also generates data keys that can be used to encrypt
and decrypt large amounts of data. Envelope encryption provided by KMS can protect your data and corresponding data keys from unauthorized access.

The following figure 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

You can use the following methods to generate a CMK:

- **Use CMKs stored in KMS**

  You can set the default server-side encryption method of the bucket to KMS without specifying a CMK ID. When sending a request to upload an object or modify the metadata of an object, you can include the `X-OSS-server-side-encryption` field in the request and set its value to KMS without specifying a CMK ID. In this method, OSS generates different keys to encrypt different objects by using the default CMK stored in KMS, and automatically decrypts the object when it is downloaded.

- **Use BYOK to implement CMKs**

  Server-side encryption supports Bring Your Own Key (BYOK). You can set the default server-side encryption method of the bucket to KMS without specifying a CMK ID. When sending a request to upload an object or modify the metadata of an object, you can include the `X-OSS-server-side-encryption` field in the request and set its value to KMS, and set `X-OSS-server-side-encryption-key-id` to a CMK ID. OSS uses the specified CMK to generate different keys to encrypt different objects, and records the CMK ID of the encrypted object to the metadata of the object. When a user who has
decryption permissions downloads the object, OSS automatically decrypts the object.

You can import your BYOK material into KMS as the CMK as follows:

- **BYOK material provided by Alibaba Cloud:** When creating a key on KMS, you can select Alibaba Cloud KMS as the source of the key material.
- **BYOK material provided by the user:** When creating a key on KMS, you can select the source of the key material as external and import external key material as required. For more information about how to import key material, see #unique_346.

**Notice:**

- This feature is in public preview. To obtain the related permissions, contact technical support personnel.
- Objects encrypted using BYOK cannot be copied to buckets in other regions.
- If you use a CMK to encrypt an object, the data key used for encryption is also encrypted and is stored as the object metadata.
- Server-side encryption that uses the default CMK stored in KMS only encrypts the data in the object. The metadata of the object is not encrypted.
- To use a RAM user account to encrypt objects through a specified CMK, you must grant the relevant permissions to the RAM user account. For more information, see #unique_347.

- **Server-side encryption through OSS-managed keys**

In this method, OSS generates and manages the keys used for data encryption, and provides strong and multi-factor security measures to protect data. OSS server-side encryption uses AES-256, one of the strongest block ciphers available, to encrypt your data.

This encryption method is an attribute of objects. To perform server-side encryption on an object, you can set the default server-side encryption method of the bucket to KMS without specifying a CMK ID. When sending a request to upload an object or modify the metadata of an object, you can include the X-OSS-server-side-encryption field in the request and set its value to AES256.
Permissions

To use server-side encryption with a RAM user account, you must have the following permissions:

- To configure the default encryption method for a bucket, you must have:
  - The management permission on the bucket.
  - The permission to perform the PutBucketEncryption and GetBucketEncryption operations.
  - The permission to perform the ListKeys, ListAliases, ListAliasesByKeyId, and DescribeKeys operations when you use a specified CMK ID to encrypt data.

  The RAM policy that specifies the permissions associated with CMK IDs is as follows:

  ```json
  {
    "Version": "1",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "kms:List*",
          "kms:DescribeKey"
        ],
        "Resource": [
          "acs:kms::*:1416614965936597:*" //This example allows the user to use all CMKs under the account. To restrict the user to use a CMK, enter the CMK ID.
        ]
      }
    ]
  }
  ```

- To upload an object to a bucket with the default encryption method configured, you must have:
  - The permission to upload objects to the bucket.
  - The permission to perform the ListKeys, ListAliases, ListAliasesByKeyId, DescribeKeys, and GenerateDataKey operations when you use a specified CMK ID to encrypt data. Otherwise, the object fails to be uploaded. The RAM policy that specifies the permissions associated with CMK IDs is as follows:

  ```json
  {
    "Version": "1",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "kms:List*",
          "kms:DescribeKey",
          "kms:GenerateDataKey"
        ],
        "Resource": ["acs:kms::*:1416614965936597:*"]
      }
    ]
  }
  ```
• To download an object from a bucket with the default encryption method configured, you must have:

- The permission to access objects in the bucket.
- The permission to perform the Decrypt operation when you use a specified CMK ID to decrypt data. Otherwise, the object fails to be downloaded. The RAM policy that specifies the permissions associated with CMK IDs is as follows:

```json
{
    "Version": "1",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["kms:Decrypt"],
            "Resource": [
                "acs:kms:***:1416614965936597:*"//This example indicates that the RAM user has the permission to decrypt data by using all CMKs. To decrypt data by using a specified CMK, enter the CMK ID.
            ]
        }
    ]
}
```
16 Static website hosting

16.1 Configure static website hosting

You can call `PutBucketWebsite` to configure static website hosting for your bucket and access the static website through the OSS domain name of the bucket.

Notice:

- Starting from August 13, 2018, `Content-Disposition: 'attachment=filename';` will be added to the response header when you access webpage objects stored in a Mainland China region or China (Hong Kong) through an OSS domain name. Webpage objects will be downloaded as attachments. Webpage objects of the following MIME types (TEXT or HTML) will be affected: HTM, HTML, JSP, PLG, HTX, and STM. This adjustment will not affect webpage objects accessed through custom domain names. We recommend that you bind a custom domain name to your bucket and use this custom domain name to access hosted websites. For more information, see Bind a custom domain.

- For more information about the API operation of configuring static website hosting, see #unique_350.

After the configuration takes effect, the OSS domain name to access a static website hosted in a bucket located in China (Hangzhou) is as follows:

http://<Bucket>.oss-cn-hangzhou.aliyuncs.com/

OSS provides the following features to manage static websites hosted in OSS:

- Index document support

  An index document links to a default index page returned by OSS when a user directly accesses the root domain name of a static website. The index page functions similar to index.html. To configure static website hosting for your bucket, you must specify an index document.

- Error document support

  An error document links to an error page returned by OSS if HTTP 4xx-related error messages such as 404 NOT FOUND occur when a user accesses a static
website. By specifying an error document, you can provide your users with appropriate error messages.

For example, set the index document to index.html and the error document to error.html. Specify oss-sample as the bucket name. The endpoint is oss-cn-hangzhou.aliyuncs.com.


### Implementation modes

<table>
<thead>
<tr>
<th>Implementation mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#unique_84</td>
<td>A user-friendly and intuitive Web application</td>
</tr>
<tr>
<td>Java SDK</td>
<td>SDK demos in various languages</td>
</tr>
<tr>
<td>Python SDK</td>
<td></td>
</tr>
<tr>
<td>PHP SDK</td>
<td></td>
</tr>
<tr>
<td>Go SDK</td>
<td></td>
</tr>
<tr>
<td>C SDK</td>
<td></td>
</tr>
<tr>
<td>.NET SDK</td>
<td></td>
</tr>
<tr>
<td>Node.js SDK</td>
<td></td>
</tr>
<tr>
<td>Ruby SDK</td>
<td></td>
</tr>
</tbody>
</table>

### Detail analysis

- **Static websites are websites where all Webpages are composed of static content**, including scripts such as JavaScript 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 static website hosted in a bucket through a custom domain name, you can **Bind a custom domain**.
• When you set a bucket to the static website hosting mode:
  
  - The index document is required, whereas 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, make sure that the objects to be specified for the index and error documents are standard objects or restored archives. Otherwise, the static website will not be accessible.

• After you set a bucket to the static website hosting mode:
  
  - OSS returns the index document for anonymous access to the root domain name of the static website. OSS returns the result of GetBucket for authorized access to the root domain of the static website.
  - OSS returns a specified object to users who access the root domain name of a static website or an object that does not exist in OSS. OSS also charges fees for such requests and generated traffic.

More information

• Tutorial: Host a static website using a custom domain name
• #unique_359

16.2 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 examplewebsite.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://examplewebsite.com and http://www.examplewebsite.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 examplewebsite.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 examplewebsite.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 examplewebsite.com and subdomain such as http://www.examplewebsite.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.

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
<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
<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 examplewebsite.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 examplewebsite.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 examplewebsite.com, you can use this examplewebsite.com to access your OSS bucket.

Similarly, you also need to bind your subdomain www.examplewebsite.com to your OSS bucket redirectbucket, so that you can use www.examplewebsite.com instead of the OSS assigned domain name originbucket.oss-cn-beijing.aliyuncs.com to access your OSS bucket.
To bind your root domain examplewebsite.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.examplewebsite.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*.
   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:

<table>
<thead>
<tr>
<th>URL</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://examplewebsite.com">http://examplewebsite.com</a></td>
<td>Displays the index document in the originbucket.</td>
</tr>
<tr>
<td>The URL of a file that does not exist in the originbucket, for example, <a href="http://examplewebsite.com/abc">http://examplewebsite.com/abc</a></td>
<td>Displays the error document in the originbucket.</td>
</tr>
</tbody>
</table>

**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.
17 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: 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

<table>
<thead>
<tr>
<th>Solution</th>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solution 1: Bind a domain to the bucket and configure an Anti-DDoS Pro instance</td>
<td>Simple configurations: You can perform the configurations through the GUI in the console.</td>
<td>Scenarios: Only buckets that are not added to the sandbox can be protected.</td>
</tr>
<tr>
<td>Solution 2: Configure a reverse proxy by using ECS and configure an Anti-DDoS Pro instance</td>
<td>· 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.</td>
<td>· Complex configurations: You must set up an NGINX reverse proxy. · High costs: You need to purchase ECS to set up an NGINX reverse proxy.</td>
</tr>
</tbody>
</table>
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.
18 Monitoring service

18.1 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.

**18.2 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.

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.

- Bucket number: 15
- Alarm rules amount: 0
- In Alarm: 0
- Forbidden amount: 0
- Triggered: 0

- 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.

- **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, etc.
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:

![Metric Charts](image)

**Alarm Rules**

The Alarm rules tab page allows you to view and manage all your alarm rules, as shown in the following figure:

![Alarm Rules Tab](image)

For the description and usage of the "Alarm Rules" tab page, see the *Alarm Service User Guide*.
18.3 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.

• 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.
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.

- 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.

- 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.

18.4 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:

```java
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:

```java
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:

```json
{"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:

```java
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:

```java
request.setPeriod("60");
```

### Metric

The Monitoring indicators reference describes the following metric items.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Metric item name</th>
<th>Unit</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserAvailability</td>
<td>User-level availability</td>
<td>%</td>
<td>User level</td>
</tr>
<tr>
<td>UserRequestValidRate</td>
<td>User-level valid request rate</td>
<td>%</td>
<td>User level</td>
</tr>
<tr>
<td>UserTotalRequestCount</td>
<td>User-level requests</td>
<td>Times</td>
<td>User level</td>
</tr>
<tr>
<td>UserValidRequestCount</td>
<td>User-level valid requests</td>
<td>Times</td>
<td>User level</td>
</tr>
<tr>
<td>UserInternetSend</td>
<td>User-level Internet outbound traffic</td>
<td>Byte</td>
<td>User level</td>
</tr>
<tr>
<td>UserInternetRecv</td>
<td>User-level Internet inbound traffic</td>
<td>Byte</td>
<td>User level</td>
</tr>
<tr>
<td>UserIntranetSend</td>
<td>User-level intranet outbound traffic</td>
<td>Byte</td>
<td>User level</td>
</tr>
<tr>
<td>Metric</td>
<td>Metric item name</td>
<td>Unit</td>
<td>Level</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------------------------------</td>
<td>------</td>
<td>--------------</td>
</tr>
<tr>
<td>UserIntranetRecv</td>
<td>User-level intranet inbound traffic</td>
<td>Byte</td>
<td>User level</td>
</tr>
<tr>
<td>UserCdnSend</td>
<td>User-level CDN outbound traffic</td>
<td>Byte</td>
<td>User level</td>
</tr>
<tr>
<td>UserCdnRecv</td>
<td>User-level CDN inbound traffic</td>
<td>Byte</td>
<td>User level</td>
</tr>
<tr>
<td>UserSyncSend</td>
<td>User-level outbound traffic of cross-region replication</td>
<td>Byte</td>
<td>User level</td>
</tr>
<tr>
<td>UserSyncRecv</td>
<td>User-level inbound traffic of cross-region replication</td>
<td>Byte</td>
<td>User level</td>
</tr>
<tr>
<td>UserServer ErrorCount</td>
<td>User-level server-site error requests</td>
<td>Times</td>
<td>User level</td>
</tr>
<tr>
<td>UserServer ErrorRate</td>
<td>User-level server-site error request rate</td>
<td>%</td>
<td>User level</td>
</tr>
<tr>
<td>UserNetwork ErrorCount</td>
<td>User-level network-site error requests</td>
<td>Times</td>
<td>User level</td>
</tr>
<tr>
<td>UserNetwork ErrorRate</td>
<td>User-level network-site error request rate</td>
<td>%</td>
<td>User level</td>
</tr>
<tr>
<td>UserAuthorizationErrorCount</td>
<td>User-level client-site authorization error requests</td>
<td>Times</td>
<td>User level</td>
</tr>
<tr>
<td>UserAuthorizationErrorRate</td>
<td>User-level client-site authorization error request rate</td>
<td>%</td>
<td>User level</td>
</tr>
<tr>
<td>UserResourceNotFound ErrorCount</td>
<td>User-level client-site error requests indicating resource not found</td>
<td>Times</td>
<td>User level</td>
</tr>
<tr>
<td>UserResourceNotFound ErrorRate</td>
<td>User-level client-site error request rate indicating resource not found</td>
<td>%</td>
<td>User level</td>
</tr>
<tr>
<td>Metric</td>
<td>Metric item name</td>
<td>Unit</td>
<td>Level</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>-------</td>
<td>-----------</td>
</tr>
<tr>
<td>UserClient TimeoutErr</td>
<td>User-level client-site time-out error request</td>
<td>Times</td>
<td>User level</td>
</tr>
<tr>
<td>orCount</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UserClient OtherError</td>
<td>User-level client-site time-out error request rate</td>
<td>%</td>
<td>User level</td>
</tr>
<tr>
<td>Rate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UserClient OtherError</td>
<td>Other user-level client-site error requests</td>
<td>Times</td>
<td>User level</td>
</tr>
<tr>
<td>Count</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UserClient OtherError</td>
<td>Other user-level client-site error request rate</td>
<td>%</td>
<td>User level</td>
</tr>
<tr>
<td>Rate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UserSuccessCount</td>
<td>Successful user-level requests</td>
<td>Times</td>
<td>User level</td>
</tr>
<tr>
<td>UserSuccessRate</td>
<td>Successful user-level request rate</td>
<td>%</td>
<td>User level</td>
</tr>
<tr>
<td>UserRedirectCount</td>
<td>User-level redirect requests</td>
<td>Times</td>
<td>User level</td>
</tr>
<tr>
<td>UserRedirectRate</td>
<td>User-level redirect request rate</td>
<td>%</td>
<td>User level</td>
</tr>
<tr>
<td>Availability</td>
<td>Availability</td>
<td>%</td>
<td>Bucket level</td>
</tr>
<tr>
<td>RequestValidRate</td>
<td>Valid request rate</td>
<td>%</td>
<td>Bucket level</td>
</tr>
<tr>
<td>TotalRequestCount</td>
<td>Requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>ValidRequestCount</td>
<td>Valid requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>InternetSend</td>
<td>Internet outbound traffic</td>
<td>Byte</td>
<td>Bucket level</td>
</tr>
<tr>
<td>InternetRecv</td>
<td>Internet inbound traffic</td>
<td>Byte</td>
<td>Bucket level</td>
</tr>
<tr>
<td>IntranetSend</td>
<td>Intranet outbound traffic</td>
<td>Byte</td>
<td>Bucket level</td>
</tr>
<tr>
<td>IntranetRecv</td>
<td>Intranet inbound traffic</td>
<td>Byte</td>
<td>Bucket level</td>
</tr>
<tr>
<td>CdnSend</td>
<td>CDN outbound traffic</td>
<td>Byte</td>
<td>Bucket level</td>
</tr>
<tr>
<td>Metric</td>
<td>Metric item name</td>
<td>Unit</td>
<td>Level</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------</td>
<td>-------</td>
<td>------------------</td>
</tr>
<tr>
<td>CdnRecv</td>
<td>CDN inbound traffic</td>
<td>Byte</td>
<td>Bucket level</td>
</tr>
<tr>
<td>SyncSend</td>
<td>Outbound traffic of cross-region replication</td>
<td>Byte</td>
<td>Bucket level</td>
</tr>
<tr>
<td>SyncRecv</td>
<td>Inbound traffic of cross-region replication</td>
<td>Byte</td>
<td>Bucket level</td>
</tr>
<tr>
<td>ServerErrorCount</td>
<td>Server-site error requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>ServerErrorRate</td>
<td>Server-site error request rate</td>
<td>%</td>
<td>Bucket level</td>
</tr>
<tr>
<td>NetworkErrorCount</td>
<td>Network-site error requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>NetworkErrorRate</td>
<td>Network-site error request rate</td>
<td>%</td>
<td>Bucket level</td>
</tr>
<tr>
<td>AuthorizationErrorCount</td>
<td>Client-site authorization error requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>AuthorizationErrorRate</td>
<td>Client-site authorization error request rate</td>
<td>%</td>
<td>Bucket level</td>
</tr>
<tr>
<td>ResourceNotFoundErrorCount</td>
<td>Client-site error requests indicating resource not found</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>ResourceNotFoundErrorRate</td>
<td>Client-site error request rate indicating resource not found</td>
<td>%</td>
<td>Bucket level</td>
</tr>
<tr>
<td>ClientTimeOutErrorCount</td>
<td>Client-site time-out error requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>ClientTimeOutErrorRate</td>
<td>Client-site time-out error request rate</td>
<td>%</td>
<td>Bucket level</td>
</tr>
<tr>
<td>ClientOtherErrorCount</td>
<td>Other client-site error requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>ClientOtherErrorRate</td>
<td>Other client-site error request rate</td>
<td>%</td>
<td>Bucket level</td>
</tr>
<tr>
<td>Metric</td>
<td>Metric item name</td>
<td>Unit</td>
<td>Level</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------------------------------------------</td>
<td>------------</td>
<td>----------</td>
</tr>
<tr>
<td>SuccessCount</td>
<td>Successful requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>SuccessRate</td>
<td>Successful request rate</td>
<td>%</td>
<td>Bucket level</td>
</tr>
<tr>
<td>RedirectCount</td>
<td>Redirect requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>RedirectRate</td>
<td>Redirect request rate</td>
<td>%</td>
<td>Bucket level</td>
</tr>
<tr>
<td>GetObjectE2eLatency</td>
<td>Average E2E latency of GetObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>GetObjectServerLatency</td>
<td>Average server latency of GetObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>MaxGetObjectE2eLatency</td>
<td>Maximum E2E latency of GetObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>MaxGetObjectServerLatency</td>
<td>Maximum server latency of GetObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>HeadObjectE2eLatency</td>
<td>Average E2E latency of HeadObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>HeadObjectServerLatency</td>
<td>Average server latency of HeadObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>MaxHeadObjectE2eLatency</td>
<td>Maximum E2E latency of HeadObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>MaxHeadObjectServerLatency</td>
<td>Maximum server latency of HeadObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>PutObjectE2eLatency</td>
<td>Average E2E latency of PutObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>Metric</td>
<td>Metric item name</td>
<td>Unit</td>
<td>Level</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>---------------</td>
<td>---------------</td>
</tr>
<tr>
<td>PutObjectServerLatency</td>
<td>Average server latency of PutObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>MaxPutObjectE2eLatency</td>
<td>Maximum E2E latency of PutObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>MaxPutObjectServerLatency</td>
<td>Maximum server latency of PutObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>PostObjectE2eLatency</td>
<td>Average E2E latency of PostObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>PostObjectServerLatency</td>
<td>Average server latency of PostObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>MaxPostObjectE2eLatency</td>
<td>Maximum E2E latency of PostObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>MaxPostObjectServerLatency</td>
<td>Maximum server latency of PostObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>AppendObjectE2eLatency</td>
<td>Average E2E latency of AppendObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>AppendObjectServerLatency</td>
<td>Average server latency of AppendObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>MaxAppendObjectE2eLatency</td>
<td>Maximum E2E latency of AppendObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td>Metric</td>
<td>Metric item name</td>
<td>Unit</td>
<td>Level</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>--------------</td>
<td>--------------</td>
</tr>
<tr>
<td><strong>MaxAppendObjectServerLatency</strong></td>
<td>Maximum server latency of AppendObject requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td><strong>UploadPartE2eLatency</strong></td>
<td>Average E2E latency of UploadPart requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td><strong>UploadPartServerLatency</strong></td>
<td>Average server latency of UploadPart requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td><strong>MaxUploadPartE2eLatency</strong></td>
<td>Maximum E2E latency of UploadPart requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td><strong>MaxUploadPartServerLatency</strong></td>
<td>Maximum server latency of UploadPart requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td><strong>UploadPartCopyE2eLatency</strong></td>
<td>Average E2E latency of UploadPartCopy requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td><strong>UploadPartCopyServerLatency</strong></td>
<td>Average server latency of UploadPartCopy requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td><strong>MaxUploadPartCopyE2eLatency</strong></td>
<td>Maximum E2E latency of UploadPartCopy requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td><strong>MaxUploadPartCopyServerLatency</strong></td>
<td>Maximum server latency of UploadPartCopy requests</td>
<td>Millisecond</td>
<td>Bucket level</td>
</tr>
<tr>
<td><strong>GetObjectCount</strong></td>
<td>Successful GetObject requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
</tbody>
</table>
## Metric Items of Metering Indicators

The following table lists the metric items of metering indicators with an aggregation granularity of 3,600s.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Metric item name</th>
<th>Unit</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>HeadObjectCount</td>
<td>Successful HeadObject requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>PutObjectCount</td>
<td>Successful PutObject requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>PostObjectCount</td>
<td>Successful PostObject requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>AppendObjectCount</td>
<td>Successful AppendObject requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>UploadPartCount</td>
<td>Successful UploadPart requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>UploadPartCopyCount</td>
<td>Successful UploadPartCopy requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>DeleteObjectCount</td>
<td>Successful DeleteObject requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
<tr>
<td>DeleteObjectsCount</td>
<td>Successful DeleteObjects requests</td>
<td>Times</td>
<td>Bucket level</td>
</tr>
</tbody>
</table>

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.
<table>
<thead>
<tr>
<th>Metric</th>
<th>Metric item name</th>
<th>Unit</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>MeteringGetRequest</td>
<td>Get requests</td>
<td>Times</td>
<td>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.</td>
</tr>
<tr>
<td>MeteringPutRequest</td>
<td>Put requests</td>
<td>Times</td>
<td>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.</td>
</tr>
<tr>
<td>Meteringinternettx</td>
<td>Volume of Internet outbound traffic</td>
<td>Byte</td>
<td>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.</td>
</tr>
<tr>
<td>MeteringCdnTX</td>
<td>Volume of CDN outbound traffic</td>
<td>Byte</td>
<td>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.</td>
</tr>
<tr>
<td>Metric</td>
<td>Metric item name</td>
<td>Unit</td>
<td>Level</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------</td>
<td>------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>MeteringSyncRX</td>
<td>Volume of inbound traffic of cross-region replication</td>
<td>Byte</td>
<td>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.</td>
</tr>
</tbody>
</table>

Sample code written by the Java SDK:

```java
request.setMetric("UserAvailability");
```

### 18.5 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:

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>%</td>
<td>An indicator showing the system availability of using the storage service. It is obtained through the equation: 1 - percentage of requests with server-end errors (indicated by a return code 5xx) in all requests.</td>
</tr>
<tr>
<td>Valid requests rate</td>
<td>%</td>
<td>Percentage of valid requests in all requests. For more information about valid requests, see the following description.</td>
</tr>
<tr>
<td>Requests</td>
<td>Times</td>
<td>Total number of requests received and processed by the OSS server</td>
</tr>
<tr>
<td>Valid requests</td>
<td>Times</td>
<td>Total number of requests whose return code is 2xx or 3xx.</td>
</tr>
<tr>
<td>Internet outbound traffic</td>
<td>Byte</td>
<td>Downstream Internet traffic</td>
</tr>
<tr>
<td>Internet inbound traffic</td>
<td>Byte</td>
<td>Upstream Internet traffic</td>
</tr>
<tr>
<td>Intranet outbound traffic</td>
<td>Byte</td>
<td>Downstream intranet traffic of the service system</td>
</tr>
<tr>
<td>Intranet inbound traffic</td>
<td>Byte</td>
<td>Upstream intranet traffic of the service system</td>
</tr>
<tr>
<td>CDN outbound traffic</td>
<td>Byte</td>
<td>Downstream CDN traffic when CDN acceleration service is activated, that is, the origin retrieval traffic</td>
</tr>
<tr>
<td>Indicator</td>
<td>Unit</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CDN inbound traffic</td>
<td>Byte</td>
<td>Upstream CDN traffic when CDN acceleration service is activated</td>
</tr>
<tr>
<td>Outbound traffic of cross-region replication</td>
<td>Byte</td>
<td>Downstream traffic generated in the data replication process when the cross-region replication function is activated</td>
</tr>
<tr>
<td>Inbound traffic of cross-region replication</td>
<td>Byte</td>
<td>Upstream traffic generated in the data replication process when the cross-region replication function is activated</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server-site error requests</td>
<td>Times</td>
<td>Total number of requests with system-level errors indicated by a return code 5xx</td>
</tr>
<tr>
<td>Server-site error requests rate</td>
<td>%</td>
<td>Percentage of requests with server-end errors in all requests</td>
</tr>
<tr>
<td>Network error requests</td>
<td>Times</td>
<td>Total number of requests whose HTTP status code is 499</td>
</tr>
<tr>
<td>Network error requests rate</td>
<td>%</td>
<td>Percentage of requests with network errors in all requests</td>
</tr>
<tr>
<td>Indicator</td>
<td>Unit</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Client-end authorization error requests</td>
<td>Times</td>
<td>Total number of requests with a return code 403</td>
</tr>
<tr>
<td>Client-end authorization error requests rate</td>
<td>%</td>
<td>Percentage of requests with client-end authorization errors in all requests</td>
</tr>
<tr>
<td>Client-end error requests indicating resource not found</td>
<td>Times</td>
<td>Total number of requests with a return code 404</td>
</tr>
<tr>
<td>Client-end error requests rate indicating resource not found</td>
<td>%</td>
<td>Percentage of requests with client-end errors indicating resource not found in all requests</td>
</tr>
<tr>
<td>Client-end time-out error requests</td>
<td>Times</td>
<td>Total number of requests whose return status code is 408 or return OSS error is RequestTimeout</td>
</tr>
<tr>
<td>Client-end time-out error requests rate</td>
<td>%</td>
<td>Percentage of requests with client-end time-out errors in all requests</td>
</tr>
<tr>
<td>Other client-end error requests</td>
<td>Times</td>
<td>Total number of requests with other client-end errors indicated by a return code 4xx</td>
</tr>
<tr>
<td>Other client-end error requests rate</td>
<td>%</td>
<td>Percentage of requests with other client-end errors in all requests</td>
</tr>
<tr>
<td>Successful requests</td>
<td>Times</td>
<td>Total number of requests whose return code is 2xx.</td>
</tr>
<tr>
<td>Successful requests rate</td>
<td>%</td>
<td>Percentage of successful requests in all requests</td>
</tr>
<tr>
<td>Redirect requests</td>
<td>Times</td>
<td>Total number of requests whose return code is 3xx.</td>
</tr>
<tr>
<td>Redirect requests rate</td>
<td>%</td>
<td>Percentage of redirect requests in all requests</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage size</td>
<td>Byte</td>
<td>Size of the total storage occupied by all buckets of a specified user before the metering statistic collection deadline.</td>
</tr>
<tr>
<td>Internet outbound traffic</td>
<td>Byte</td>
<td>Total Internet outbound traffic of the user from 00:00 of the first day of the current month to the metering statistic collection deadline.</td>
</tr>
<tr>
<td>Put requests</td>
<td>Times</td>
<td>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.</td>
</tr>
<tr>
<td>Get requests</td>
<td>Times</td>
<td>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.</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage size</td>
<td>Byte</td>
<td>Size of storage occupied by a specified bucket before the metering statistic collection deadline.</td>
</tr>
<tr>
<td>Internet outbound traffic</td>
<td>Byte</td>
<td>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.</td>
</tr>
<tr>
<td>Put requests</td>
<td>Times</td>
<td>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.</td>
</tr>
<tr>
<td>Get requests</td>
<td>Times</td>
<td>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.</td>
</tr>
</tbody>
</table>
Metering indicators are monitored chronologically, and are collected and aggregated at the hour-level. Details of metering indicators are as follows:

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage size</td>
<td>Byte</td>
<td>Average size of storage used by a specified bucket in an hour.</td>
</tr>
<tr>
<td>Internet outbound traffic</td>
<td>Byte</td>
<td>Total Internet outbound traffic of a specified bucket in an hour.</td>
</tr>
<tr>
<td>Put requests</td>
<td>Times</td>
<td>Total number of Put requests of a specified bucket in an hour.</td>
</tr>
<tr>
<td>Get requests</td>
<td>Times</td>
<td>Total number of Get requests of a specified bucket in an hour.</td>
</tr>
</tbody>
</table>

**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:
<table>
<thead>
<tr>
<th>Indicator</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average E2E latency of GetObject requests</td>
<td>Millisecond</td>
<td>Average E2E latency of successful requests whose request API is GetObject</td>
</tr>
<tr>
<td>Average server latency of GetObject requests</td>
<td>Millisecond</td>
<td>Average server latency of successful requests whose request API is GetObject</td>
</tr>
<tr>
<td>Maximum E2E latency of GetObject requests</td>
<td>Millisecond</td>
<td>Maximum E2E latency of successful requests whose request API is GetObject</td>
</tr>
<tr>
<td>Maximum server latency of GetObject requests</td>
<td>Millisecond</td>
<td>Maximum server latency of successful requests whose request API is GetObject</td>
</tr>
<tr>
<td>Average E2E latency of HeadObject requests</td>
<td>Millisecond</td>
<td>Average E2E latency of successful requests whose request API is HeadObject</td>
</tr>
<tr>
<td>Average server latency of HeadObject requests</td>
<td>Millisecond</td>
<td>Average server latency of successful requests whose request API is HeadObject</td>
</tr>
<tr>
<td>Maximum E2E latency of HeadObject requests</td>
<td>Millisecond</td>
<td>Maximum E2E latency of successful requests whose request API is HeadObject</td>
</tr>
<tr>
<td>Maximum server latency of HeadObject requests</td>
<td>Millisecond</td>
<td>Maximum server latency of successful requests whose request API is HeadObject</td>
</tr>
<tr>
<td>Average E2E latency of PutObject requests</td>
<td>Millisecond</td>
<td>Average E2E latency of successful requests whose request API is PutObject</td>
</tr>
<tr>
<td>Average server latency of PutObject requests</td>
<td>Millisecond</td>
<td>Average server latency of successful requests whose request API is PutObject</td>
</tr>
<tr>
<td>Maximum E2E latency of PutObject requests</td>
<td>Millisecond</td>
<td>Maximum E2E latency of successful requests whose request API is PutObject</td>
</tr>
<tr>
<td>Indicator</td>
<td>Unit</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Maximum server latency of PutObject requests</td>
<td>Millisecond</td>
<td>Maximum server latency of successful requests whose request API is PutObject</td>
</tr>
<tr>
<td>Average E2E latency of PostObject requests</td>
<td>Millisecond</td>
<td>Average E2E latency of successful requests whose request API is PostObject</td>
</tr>
<tr>
<td>Average server latency of PostObject requests</td>
<td>Millisecond</td>
<td>Average server latency of successful requests whose request API is PostObject</td>
</tr>
<tr>
<td>Maximum E2E latency of PostObject requests</td>
<td>Millisecond</td>
<td>Maximum E2E latency of successful requests whose request API is PostObject</td>
</tr>
<tr>
<td>Maximum server latency of PostObject requests</td>
<td>Millisecond</td>
<td>Maximum server latency of successful requests whose request API is PostObject</td>
</tr>
<tr>
<td>Average E2E latency of AppendObject requests</td>
<td>Millisecond</td>
<td>Average E2E latency of successful requests whose request API is AppendObject</td>
</tr>
<tr>
<td>Average server latency of AppendObject requests</td>
<td>Millisecond</td>
<td>Average server latency of successful requests whose request API is AppendObject</td>
</tr>
<tr>
<td>Maximum E2E latency of AppendObject requests</td>
<td>Millisecond</td>
<td>Maximum E2E latency of successful requests whose request API is AppendObject</td>
</tr>
<tr>
<td>Maximum server latency of AppendObject requests</td>
<td>Millisecond</td>
<td>Maximum server latency of successful requests whose request API is AppendObject</td>
</tr>
<tr>
<td>Average E2E latency of UploadPart requests</td>
<td>Millisecond</td>
<td>Average E2E latency of successful requests whose request API is UploadPart</td>
</tr>
</tbody>
</table>
### Monitoring service indicator

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average server latency of UploadPart requests</td>
<td>Millisecond</td>
<td>Average server latency of successful requests whose request API is UploadPart</td>
</tr>
<tr>
<td>Maximum E2E latency of UploadPart requests</td>
<td>Millisecond</td>
<td>Maximum E2E latency of successful requests whose request API is UploadPart</td>
</tr>
<tr>
<td>Maximum server latency of UploadPart requests</td>
<td>Millisecond</td>
<td>Maximum server latency of successful requests whose request API is UploadPart</td>
</tr>
<tr>
<td>Average E2E latency of UploadPartCopy requests</td>
<td>Millisecond</td>
<td>Average E2E latency of successful requests whose request API is UploadPart Copy</td>
</tr>
<tr>
<td>Average server latency of UploadPartCopy requests</td>
<td>Millisecond</td>
<td>Average server latency of successful requests whose request API is UploadPart Copy</td>
</tr>
<tr>
<td>Maximum E2E latency of UploadPartCopy requests</td>
<td>Millisecond</td>
<td>Maximum E2E latency of successful requests whose request API is UploadPart Copy</td>
</tr>
<tr>
<td>Maximum server latency of UploadPartCopy requests</td>
<td>Millisecond</td>
<td>Maximum server latency of successful requests whose request API is UploadPart Copy</td>
</tr>
</tbody>
</table>

#### 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:

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
</table>
| Successful GetObject requests     | Times | Number of successful requests whose request API is GetObject }
### Indicator | Unit | Description
---|---|---
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
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

### 18.6 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*.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Sum value</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>user authorization error count</td>
<td>12vunjankong.metric.urn/thName.frequency</td>
<td>14.29%</td>
</tr>
<tr>
<td>user success count</td>
<td>72vunjankong.metric.urn/thName.frequency</td>
<td>85.71%</td>
</tr>
<tr>
<td>Sum</td>
<td>84vunjankong.metric.urn/thName.frequency</td>
<td>100%</td>
</tr>
</tbody>
</table>

**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

![Graphs showing successful request categories](Image)
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 significantly, 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:

![Monitoring Statistics Chart](image)

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 issue.
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 getRequestIds 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 OSSException’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
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 long to investigate the client application, or use a network tool such as Wireshark to check your network for problems.
19 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.
20 Event notification

OSS allows you to configure event notification to learn about related operations on resources in a timely manner.

When you create an event notification rule, you can customize the event notification rule. This way, you can be notified of specified object operations in OSS in a timely manner. Examples:

- New data has been uploaded from the picture content sharing platform, audio and video platform to OSS.
- Related content on OSS has been updated.
- The important files on the OSS are deleted.
- The synchronization of data on OSS has been completed.

Notice:

Notifications are not sent for the ts and m3u8 files generated in RTMP ingesting.

Event notification in OSS is asynchronous and does not affect normal OSS operations. To configure event notification, you must configure the rule and message notification.

- Rule: specifies conditions for OSS resource usage for which to send notifications.
- Message notification: relies on Alibaba Cloud MNS to provide multiple notification methods.

The following figure shows the overall architecture of event notification for OSS resources.
Implementation modes

For more information about the configurations of event notification, see #unique_373.

Event types

<table>
<thead>
<tr>
<th>Event type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ObjectCreated:PutObject</td>
<td>Created/overwritten an object: simple upload.</td>
</tr>
<tr>
<td>ObjectCreated:PostObject</td>
<td>Created/overwritten object: form upload.</td>
</tr>
<tr>
<td>ObjectCreated:CopyObject</td>
<td>Created/overwritten an object: object copy.</td>
</tr>
<tr>
<td>ObjectCreated:InitiateMultipartUpload</td>
<td>Created/overwritten an object: A multipart upload task was initialized.</td>
</tr>
<tr>
<td>ObjectCreated:UploadPart</td>
<td>Created/overwritten an object: Parts were uploaded.</td>
</tr>
<tr>
<td>ObjectCreated:UploadPartCopy</td>
<td>Created/overwritten an object: Parts were uploaded by copying data from an existing object.</td>
</tr>
<tr>
<td>ObjectCreated:CompleteMultipartUpload</td>
<td>Created/overwritten an object: A multipart upload task was completed.</td>
</tr>
<tr>
<td>ObjectCreated:AppendObject</td>
<td>Created/overwritten an object: append upload.</td>
</tr>
<tr>
<td>ObjectDownloaded:GetObject</td>
<td>Downloaded an object: simple download.</td>
</tr>
<tr>
<td>ObjectRemoved:DeleteObject</td>
<td>An object was deleted.</td>
</tr>
</tbody>
</table>
## Event notification

<table>
<thead>
<tr>
<th>Event type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>**ObjectRemoved:**DeleteObjects</td>
<td>Multiple objects were deleted.</td>
</tr>
<tr>
<td>**ObjectReplication:**ObjectCreated</td>
<td>Synchronized an operation on an object: The object was created.</td>
</tr>
<tr>
<td>**ObjectReplication:**ObjectRemoved</td>
<td>Synchronized an operation on an object: The object was deleted.</td>
</tr>
<tr>
<td>**ObjectReplication:**ObjectModified</td>
<td>Synchronized an operation on an object: The object was overwritten.</td>
</tr>
</tbody>
</table>

### Notification message formats

The message content of OSS-based event notification is encoded in Base64. The decoded content is in JSON format. Example:

```json
{"events": [{
    "eventName": "",
    "eventSource": "",
    "eventTime": "",
    "eventVersion": "",
    "oss": {
        "bucket": {
            "arn": "", //The ARN for the bucket. The format is "acs:oss:region:uid:bucket".
            "name": "", //The name of the bucket.
            "ownerIdentity": ""}, //The owner of the bucket.
        "object": {
            "deltaSize": , //The changed size of the object. If you create an object, this value is the size of the object. If you overwrite an existing object, this value is the difference between the new and existing objects, which may be a negative value.
            "eTag": "", //The ETag of the object. This value is the same as the ETag header value in the response to the GetObject() request.
            "key": "", //The name of the object.
            "position": , //Required only for the ObjectCreated: AppendObject event. This parameter indicates the position calculated based on the size of the object last uploaded. The value starts from 0.
            "readFrom": , //Required only for the ObjectDownloaded: GetObject event. This parameter indicates the beginning of the range of Bytes to read data from the object. The value starts from 0 for the range request. Otherwise, the value is 0.
            "readTo": , //Required only for the ObjectDownloaded: GetObject event. This parameter indicates the ending of the range of Bytes to read data from the object. The value increases by one for the ending Byte in the range request. Otherwise, the value is the actual size of the object.
            "size": }, //The size of the object.
            "ossSchemaVersion": "", //The version of the schema. The value is 1.0.
            "ruleId": "GetObject"}], //The rule ID that matches the event.```
"region": "", // The region in which the bucket is located.
"requestParameters": {
  "sourceIP Address": " ", // The source IP address in the request.
"responseElements": {
  "requestId": " ", // The ID of the request.
"userIdentity": {
  "principalId": " ", // The user ID of the requester.
"xVars": { // The custom parameters in the callback of OSS.
  "x:callback-var1": "value1",
  "x:callback-var2": "value2"}}}

Examples:

{"events": [{
  "eventName": "ObjectDownloaded:GetObject",
  "eventSource": "acs:oss",
  "eventTime": "2016-07-01T11:17:30.000Z",
  "eventVersion": "1.0",
  "oss": {
    "bucket": {
      "arn": "acs:oss:cn-shenzhen:11489********46818:event-notification-test-shenzhen",
      "name": "event-notification-test-shenzhen",
      "ownerIdentity": "11489********46818"},
    "object": {
      "deltaSize": 0,
      "eTag": "0CC175B9C0F1B6xxxxxx99E269772661",
      "key": "test",
      "readFrom": 0,
      "readTo": 1,
      "size": 1},
    "ossSchemaVersion": "1.0",
    "ruleId": "GetObjectRule"},
  "region": "cn-shenzhen",
  "requestParameters": {
    "sourceIP Address": "140.xx.xx.90"},
  "responseElements": {
    "requestId": "5776514Axxxxxxx542425D2B"},
  "userIdentity": {
    "principalId": "11489********46818"},
  "xVars": {
    "x:callback-var1": "value1",
    "x:callback-var2": "value2"}}]}

21.1 Access control

21.1.1 Overview

Alibaba Cloud’s permission management mechanism includes Resource Access Management (RAM) and Security Token Service (STS). This enables users to access OSS through subaccounts with different permissions and grants users temporary access authorization. Usage of RAM and STS can greatly improve management flexibility and security.

The following content is introduced in permission management:

- What is RAM and STS
- Access a bucket without using the primary account
- Read/Write permission separation
- Access control
- STS temporary access authorization
- The problem of OSS authority and Its Troubleshooting
- STS frequently asked questions and troubleshooting
- OSS sub-account setup Frequently Asked Questions

Click RAM Policy Editor Online Editing allows you to generate authorization policies.

21.1.2 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’s 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 credential
s 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 osscmd, a tool in the OSS PythonSDK that allows you to directly work on OSS through the command line. osscmd can be obtained from PythonSDK.

Typical osscmd usage:

```
Download files
./osscmd 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 corresponding to your own account.

Upload files

```
./osscmd put LOCALFILE oss://BUCKET/OBJECT --host=Endpoint -i AccessKeyId -k AccessKeySecret
```

The meaning of each field is the same as for the download example.

### 21.1.3 Access a bucket without using the primary account

Assume that the user is a mobile developer and currently only has one bucket, ram-test-dev, for development, testing, and other functions. The user must stop using the primary account to access this bucket. This can avoid problems caused by AccessKey and password leaks. In the following example, replace AccessKey with your own AccessKey. The procedure is as follows:


   ![Note]

   The service must be activated first if you have never used it before.

2. Click Users to go to the User Management page.

3. The page shows that no user is created. Click New User on the upper right corner to create a subaccount with the same OSS access permissions as the primary account. Remember to select the Auto generate AccessKey for this user.

4. The AccessKey for this account is generated and must be saved for later use.

5. Return to User Management interface, which shows the newly created account named ram_test. When created, this subaccount does not have any permissions yet. Click the Authorize link on the right side and grant this subaccount full access permissions for OSS.

   After authorization, click the Management link on the right side if you want to give the subaccount console logon or other permissions.

Now we can test the uploading and downloading operations. In the example, the AccessKey is ram_test’s AccessKey. During the test, replace this with your own AccessKey.

```
$.osscmd get
oss://ram-test-dev/test.txt test.txt --host=oss-cn-hangzhou.aliyuncs.com -i oOhue*****Frogv -k OmVwFJO3qCT0*****FhOYpg3p0KnA
100% The object test.txt is downloaded to test.txt, please check.
```
0.069(s) elapsed

$.osscmd put test.txt oss://ram-test-dev/test.txt --host=oss-cn-hangzhou.aliyuncs.com -i oOhue******Frogv -k OmVwFJO3qcT0******

PhOYpg3p0KnA
100%

Object URL is: http://ram-test-dev.oss-cn-hangzhou.aliyuncs.com/test.txt
Object abstract path is: oss://ram-test-dev/test.txt
ETag is "E27172376D49FC609E7F46995E1F808F"

0.108(s) elapsed

As you can see, this subaccount can basically be used for all operations, so you can avoid leaking the primary account’s AccessKey.

21.1.4 Read/Write permission separation

When the users want to use an application server to provide external service, OSS can store back-end static resources. In this case, we recommend that the application server be granted the OSS read-only permission to reduce the risk of attacks. The read and write permission separation can be configured to grant the application server a user with the read-only permission.

1. Create an account ram_test_pub. As shown in the following figure, select ReadOnly in the authorization management area:

2. You can now use the AccessKey of the subaccount to test the upload and download permissions. The AccessKey here is a ram_test_pub AccessKey and is to be replaced with your own AccessKey during the test.

$.osscmd get oss://ram-test-dev/test.txt test.txt --host=oss-cn-hangzhou.aliyuncs.com -i oOhue******Frogv -k OmVwFJO3qcT0******

PhOYpg3p0KnA
100% The object test.txt is downloaded to test.txt, please check.

0.070(s) elapsed

$.osscmd put test.txt OSS: // Ram-test-dev/test.txt -- Host = porteroohue ***** frogv-K OmVwFJO3qcT0 * PhOYpg3p0KnA?

100% Error Headers:
[['content-length', '229'], ('server', 'AliyunOSS'), ('connection', 'keep-alive'), ('x-oss-request-id', '5646E49C1790CF0F531BAE0D'), ('date', 'Sat, 14 Nov 2015 07:37:00 GMT'), ('content-type', 'application/xml')]

Error Body:
<? xml version="1.0" encoding="UTF-8"? >
<Error>
  <Code>AccessDenied</Code>
  <Message>AccessDenied</Message>
  <RequestId>5646E49C1790CF0F531BAE0D</RequestId>
  <HostId>ram-test-dev.oss-cn-hangzhou.aliyuncs.com</HostId>
</Error>

Error Status:
403
With reference to the preceding example, we can conclude that the ram_test_pub account cannot be used to upload files.

21.1.5 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.
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:

```json
{
  "Version": "1",
  "Statement": [
    {
      "Effect": "allow",
      "Action": [
        "oss:ListObjects",
        "Oss: GetObject"
      ],
      "Resource": [
        "acs:oss:*:*:ram-test-app",
        "acs:oss:*:*:ram-test-app/*"
      ]
    }
  ]
}
```
This indicates read-only permission for ram-test-app.

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

Authorization policy format definition
Authorization policy FAQs
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:

```json
{
  "Version": "1",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "oss:DeleteObject",
        "oss:ListParts",
        "oss:AbortMultipartUpload",
        "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.

```
{
  "Statement": [
    {
      "Action": "sts:AssumeRole",
      "Effect": "Allow",
      "Resource": "acs:ram::1862122263:role/RoleName"
    }
  ],
  "Version": "1"
}
```
Create another custom authorization policy using the same method:

```
{
    "Statement": [
        {
            "Action": "sts:AssumeRole",
            "Effect": "Allow",
            "Resource": "acs:ram::1894xxxxxx722283:role/ramtestappwrite"
        }
    ],
    "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::1894xxxxxx722283:role/ramtestappreadonly RoleSessionName=usr001 Policy='{"Version":"1","Statement":[{"Effect":"Allow","Action":["oss:ListObjects","oss:GetObject"],"Resource":["acs:oss:*:*:ram-test-app","acs:oss:*:*:ram-test-app/*"]}]}' DurationSeconds=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.

```bash
[admin@NGIS-CWWF344M01C /home/admin/oss_test]
$.osscmd get oss://ram-test-app/test.txt test.txt --host=oss-cn-hangzhou.aliyuncs.com -i oOhue******Frogv -k OmVwFJO3qcT0******FhOYpg3p0KnA
Error Headers:
[('content-length', '229'), ('server', 'AliyunOSS'), ('connection', 'keep-alive'), ('x-oss-request-id', '564A94D444F4D8B2225E4AFE'), ('date', 'Tue, 17 Nov 2015 02:45:40 GMT'), ('content-type', 'application/xml')]
Error Body:
<? xml version="1.0" encoding="UTF-8"? >
<Error>
  <Code>AccessDenied</Code>
  <Message>AccessDenied</Message>
  <RequestId>564A94D444F4D8B2225E4AFE</RequestId>
  <HostId>ram-test-app.oss-cn-hangzhou.aliyuncs.com</HostId>
</Error>
Error Status: 403
get Failed!

[admin@NGIS-CWWF344M01C /home/admin/oss_test]
$.osscmd put test.txt oss://ram-test-app/test.txt --host=oss-cn-hangzhou.aliyuncs.com -i oOhue******Frogv -k OmVwFJO3qcT0******FhOYpg3p0KnA
100% Error Headers:
[('content-length', '229'), ('server', 'AliyunOSS'), ('connection', 'keep-alive'), ('x-oss-request-id', '564A94E5B1119B445B9F8C3A'), ('date', 'Tue, 17 Nov 2015 02:45:57 GMT'), ('content-type', 'application/xml')]
Error Body:
<? xml version="1.0" encoding="UTF-8"? >
<Error>
  <Code>AccessDenied</Code>
  <Message>AccessDenied</Message>
  <RequestId>564A94E5B1119B445B9F8C3A</RequestId>
  <HostId>ram-test-app.oss-cn-hangzhou.aliyuncs.com</HostId>
</Error>
Error Status: 403
```
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-CWWF344M01C /home/admin/oss_test]
   $python ./sts.py AssumeRole RoleArn=acs:ram::1894xxxxxx722283:role/ramtestappreadonly RoleSessionName=usr001 Policy='{"Version":"1","Statement": [{"Effect":"Allow","Action": ["oss:ListObjects","oss:GetObject"], "Resource": ["acs:oss:*:*:ram-test-app","acs:oss:***:ram-test-app/*"]}]'} --id=oOhue******Frogv --secret=OmwfFJO3qcT0******FhOYpg3p0KnA

   https://sts.aliyuncs.com/?SignatureVersion=1.0&Format=JSON&Timestamp=2015-11-17T03%3A07%3A25Z&RoleArn=acs%3Aram%3A%3A1894xxxxxx722283%3Arole%2Framtestappreadonly&RoleSessionName=usr001&AccessKeyId=oOhue******Frogv&Policy=%7B%22Version%22%3A%222%22%22Statement%22%3A%222%22%22Role%22%3A%222%22%22Resource%22%3A%222%22%22AccessKeyId%22%3A%222%22%22Policy%22%3A%222%22%22Expiration%22%3A%222%22%22SecurityToken%22%3A%222%22%22RequestId%22%3A%222%22%7D&SignatureMethod=HMAC-SHA1&Version=2015-04-01&Signature=bsxhPzpwRj5vch3SjaFIXLodwq9%3D&Action=AssumeRole&SignatureNonce=53e1be9c-8cd8-11e5-9b86-008cfa5e4938

   {"AssumedRoleUser": {
      "Arn": "acs:ram::1894xxxxxx722283:role/ramtestappreadonly/ usr001",
      "AssumedRoleId": "317446347657426289:usr001"
    },
    "Credentials": {
      "AccessKeyId": "STS. 3mQEbNf******wa180Le",
      "SecretKey": "B1w7rCr94dzSwnYJ******3PiPqKZ3gjQhAx6mB",
      "Expiration": "2015-11-17T04:07:25Z",
      "SecurityToken": "CAEsVAMIARAAASQ*********7683CGLhdGsv2/di8u1+X*********dxmSFTd0fpp5wpPK/TUctYH2M3//c4yMN1PUCcEHI1zppCNmpDG2XeNA30S16Jsw6ESm1S50syWwBmsYkCJW1igXnhHZ/0K+Sm1bYxLfb33q4fgCFe97fjejuej8RMgqFx0Hny2Bz9hTTFvMu0r1RWWJ0zr5YzlIT3dhgTGUiejMnxNqOJNj9mZvY1NzQvNnjl4OSGdXNyMDAXMjTRgJ2lKjo6UnNhtUQ1QpsBcExGpUBCvB6g******CgxBY3rp2b5FcvXbHMSBkfJd6lVbhogCg9vc3M6TGlzedE91amVjDjHMKDW9zs czpHZXRPympyL3QSUg0UmVzb3VvyY2VFcVXbHMSFCJlc291cmNLgjVKGgfj czpv3cM6Kjq0On3hB10ZmN8LWFwAoaYWNzO9czoq******FLtXRlc3QYXZBlypKEDE40TQoO9d3Nj3kJMjYODNStI2ODQyWg98c3Nj1bWVkm9sz2VvZ CzJgAGoSMZeN3Q2MzQ3NjU3NDI2Mjg5chJyYW10ZXN0YXJvcmVHZG9ubHk="
    },
    "RequestId": "8C09F64-F19D-4EC1-A3AD-7A718CD0849B"
2. Use the temporary credential to download files. Here sts_token is the SecurityToken returned by the STS.

```bash
[admin@NGIS-CWWF344M01C /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 zGwNYjzd postingsby.com -s sts_token=CAEvAMJRArAASQQ****** 7683CGndGsv2/di8uI+X******DxM5FTd0f6g5wpPK/7uctYH2Y3M2 //c4yMN1PUCc EHI1zzpCINmpDg2xnaZeA30S16w6E5mI50s8yWbMsYkCJW15xnhfz/OK+QSp1bYxlFB 33qfgCFe97Ijeuj8RmgfQx0Hny2BzGhHTVFMuM21RWOJZnR5YzI1t1d4MtG tGUieijMxNzQOjNzYQyNyI4OSogdXNydMAdxMjTJgrGj2KjoGUnNhtUQ 1pqsBCgXipuBcGVBb******CgXbY3RpB25FexVhMBSSkFjdGlvbhogC9vc3M6 TGlzldE9amVjDHDKV2ZczzpHZXRPyMyLyQUSugo0UmVz3y2Y2VFCxVhMBMS CFJ1lc291cmN1LgjKGGfjczpqvcz36MxKjqu0N3hj0S10ZXN0LWFwCAoaaWNe5Om9zcczqoq ********FtLXRC3QyXBlwLyPkJEDE4O7QxO3k3j4k3mjIyODNSB1I20DQyWg9Bc31bWVsKUm9s5ZVzZXJgAAsME3N0DQzQ3NjU3N1I2Mj5chJyYW10ZXN0YXhBcm 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.

```bash
[admin@NGIS-CWWF344M01C /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 zGwNYjzd postingsby.com -s sts_token=CAEvAMJRArAASQQ****** 7683CGndGsv2/di8uI+X******DxM5FTd0f6g5wpPK/7uctYH2Y3M2 //c4yMN1PUCc EHI1zzpCINmpDg2xnaZeA30S16w6E5mI50s8yWbMsYkCJW15xnhfz/OK+QSp1bYxlFB 33qfgCFe97Ijeuj8RmgfQx0Hny2BzGhHTVFMuM21RWOJZnR5YzI1t1d4MtG tGUieijMxNzQOjNzYQyNyI4OSogdXNydMAdxMjTJgrGj2KjoGUnNhtUQ 1pqsBCgXipuBcGVBb******CgXbY3RpB25FexVhMBSSkFjdGlvbhogC9vc3M6 TGlzldE9amVjDHDKV2ZczzpHZXRPyMyLyQUSugo0UmVz3y2Y2VFCxVhMBMS CFJ1lc291cmN1LgjKGGfjczpqvcz36MxKjqu0N3hj0S10ZXN0LWFwCAoaaWNe5Om9zcczqoq ********FtLXRC3QyXBlwLyPkJEDE4O7QxO3k3j4k3mjIyODNSB1I20DQyWg9Bc31bWVsKUm9s5ZVzZXJgAAsME3N0DQzQ3NjU3N1I2Mj5chJyYW10ZXN0YXhBcm VhZG9ubhk= 100% The object test.txt is downloaded to test.txt, please check.

0.061(s) elapsed
```

```
100% Error Headers:
['content-length', '254'], ('server', 'AliyunOSS'), ('connection', 'keep-alive'), ('x-oss-request-id', '564A9A2A1790CF0F53C15C82'), ('date', 'Tue, 17 Nov 2015 03:08:26 GMT'), ('content-type', 'application/xml')]

Error Body: <? xml version="1.0" encoding="UTF-8"? >
<Error>
  <Code>AccessDenied</Code>
  <Message>Access denied by authorizer's policy.</Message>
  <RequestId>564A9A2A1790CF0F53C15C82</RequestId>
  <HostId>ram-test-app.oss-cn-hangzhou.aliyuncs.com</HostId>
</Error>
```

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-CWWF344M01C /home/admin/oss_test]
   $python ./sts.py AssumeRole RoleArn=acs:ram::1894xxxx722283:role/ramtestappwrite RoleSessionName=usr001 Policy='{"Version":"1","Statement": [{"Effect":"Allow","Action":["oss:PutObject"],"Resource":["acs:oss:::ram-test-app/usr001/*"]}]}' --id=oOhue*****Frogv --secret=OmWvFJO3qcT0*****Fh0Ypg3p0KnA
   https://sts.aliyuncs.com/?SignatureVersion=1.0&Format=JSON&Timestamp=2015-11-17T03%3A16%3A10Z&RoleArn=acs%3Aram%3A%3A1894xxxx722283%3Arole%2Framtestappwrite&RoleSessionName=usr001&AccessKeyId=oOhu*****Frogv&Policy=%7B%22Version%22%3A%221%22%2C%22Statement%22%3A%22%22%2C%22Action%22%3A%22%22%22%2C%22Resource%22%3A%22%22%22%22acs%3As%2a%22%22acs%3Aosss%3A%2a%22%22acs%3Aosss%3A%2a%22RoleSessionName=usr001&AccessKeyId=STS.rtfx13******NlIJlS4U&Policy=%7B%22Version%22%3A%221%22%2C%22Statement%22%3A%22%22%2C%22Action%22%3A%22%22%22%2C%22Resource%22%3A%22%22%22%22acs%3As%2a%22%22acs%3Aosss%3A%2a%22&Expiration="2015-11-17T04:16:10Z","SecurityToken":"CAESkwMIARKAAUh3/Uzcg13******y0ZjGewMp
g311TcBeBFu1e0/3Spuidid+GsV+0lvuVXjn******a8zKJktzV0okSy+ mwUrxE5vSuvRVDntrs78csNfw0OJUMKjLxVdxWnGi1pgxJCBzNhZsYV/6ycTaTseYSE1v6kqQ7A+GpWyWwLdpGhYTFVMcumRmeDezRFLNvjWjTmxSjSmxTNFiUE jMINTQwNZg0NzY2MDAYTOQy0CoGdxXNyMDAXMOPOzoJ2RkJoGUnNHuTUQIn7YKA TcEacQoFQwxbs3c5JwoMQWnoa0WuRxFL1wyWzegZBY3Rpb24aDwoNBn3b0Lb1d E9iamVjDBi/Cg5SZXNvdXJjZUVxWdFscxIIUmVzb3VYU2aIw0wYWNW0n9czq oqOio6cmFtLXRLc3Qt0tQvZcVzjcAwMS5aqShAx0DD0kMTg5NzY5NzY5Mjg3UgUyNju g0Ml0POQXnzdw1Zf3VjVbGvvC2vYABqEjMINTQwNZg0NzY2MDAYOTQy0HIPvcm FtdGvzdGfwchDhyaxRXL }, "AssumedRoleId": "355407847660029428:usr001", }, "AssumedRoleUser": { "Arn": "acs:ram::1894xxxx722283:role/ramtestappwrite/usr001", "AssumedRoleSessionName": "usr001" }, "Credentials": { "AccessKeyId": "STS.rtfx13******NlIJlS4U", "AccessKeySecret": "2fsaM8E2maB2dn******wpsKTyK4ajo7TxFr0zIM", "Expiration": "2015-11-17T04:16:10Z", "SecurityToken": "CAESkwMIARKAAUh3/Uzcg13******y0ZjGewMp
g311TcBeBFu1e0/3Spuidid+GsV+0lvuVXjn******a8zKJktzV0okSy+ mwUrxE5vSuvRVDntrs78csNfw0OJUMKjLxVdxWnGi1pgxJCBzNhZsYV/6ycTaTseYSE1v6kqQ7A+GpWyWwLdpGhYTFVMcumRmeDezRFLNvjWjTmxSjSmxTNFiUE jMINTQwNZg0NzY2MDAYTOQy0CoGdxXNyMDAXMOPOzoJ2RkJoGUnNHuTUQIn7YKA TcEacQoFQwxbs3c5JwoMQWnoa0WuRxFL1wyWzegZBY3Rpb24aDwoNBn3b0Lb1d E9iamVjDBi/Cg5SZXNvdXJjZUVxWdFscxIIUmVzb3VYU2aIw0wYWNW0n9czq oqOio6cmFtLXRLc3Qt0tQvZcVzjcAwMS5aqShAx0DD0kMTg5NzY5NzY5Mjg3UgUyNju g0Ml0POQXnzdw1Zf3VjVbGvvC2vYABqEjMINTQwNZg0NzY2MDAYOTQy0HIPvcm FtdGvzdGfwchDhyaxRXL }, "RequestId": "19407705-54B2-41AD-AAF0-FE87E8870B0D" }

2. Let us test if we can use the credentials to upload and download.

   [admin@NGIS-CWWF344M01C /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 2fsaM8E2maB2dn******wpsKTyK4ajo7TxFr0zIM --sts_token=CAESkwMIARKAAUh3/Uzcg13******y0ZjGewMp
g311TcBeBFu1e0/3Spuidid+GsV+0lvuVXjn******a8zKJktzV0okSy+ mwUrxE5vSuvRVDntrs78csNfw0OJUMKjLxVdxWnGi1pgxJCBzNhZsYV/6ycTaTseYSE1v6kqQ7A+GpWyWwLdpGhYTFVMcumRmeDezRFLNvjWjTmxSjSmxTNFiUE jMINTQwNZg0NzY2MDAYTOQy0CoGdxXNyMDAXMOPOzoJ2RkJoGUnNHuTUQIn7YKA TcEacQoFQwxbs3c5JwoMQWnoa0WuRxFL1wyWzegZBY3Rpb24aDwoNBn3b0Lb1d E9iamVjDBi/Cg5SZXNvdXJjZUVxWdFscxIIUmVzb3VYU2aIw0wYWNW0n9czq oqOio6cmFtLXRLc3Qt0tQvZcVzjcAwMS5aqShAx0DD0kMTg5NzY5NzY5Mjg3UgUyNju g0Ml0POQXnzdw1Zf3VjVbGvvC2vYABqEjMINTQwNZg0NzY2MDAYOTQy0HIPvcm FtdGvzdGfwchDhyaxRXL

Error Headers:
The test.txt upload fails. We have formatted the entered policy discussed at the beginning of this document, which is as follows:

```json
{
    "Version": "1",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["oss:PutObject"],
            "Resource": ["acs:oss:::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.

```bash
[admin@NGIS-CWWF344M01C /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
2fsaM8E2maB2dn******wpsKTyK4ajo7TxFr0zIM --sts_token=CAESkwMIAR
KAAUh3/Uzcg13******y0IZjGewMpg31ITxClBeBFU1eO/3Spudpид+GVs+0lvu1vXJn
********a8azKJKtzV0oKSx+mwUrXsvUSRVDntrs78CsnW0OJUJKjIldWsnG1
pgx3CBzN2ZYY/6ycTaZYSSE1V6kqQ7A+GpWY******LpdGhTVFMuncRmeDe2R
FLNvWjTmxJSmxTNFU1EjM1NTQwNzg0Ny2MDAyOThq0CoGdxNyxMDAxMOPzo
J2RkjoGuNnHTU1QnYKATEcqOFOQWxsb3cSJWoMQWOaawoRXF1YWxzegZBY
3Rp24aDwoNbc3Nz01B1de9iamVjDBI/Gc55ZNxvdXjJZUxvDFsxUIUmVzfb3
VY2uwaIwohWNN0m9czqo6cFtLXRlc3Qt********VzcgAwMS8qShAxODk0MT
5g5zY5NzIyMjgzuUyNnjg6Mt0PQXndw12FJvBGVCv2VyaYAbxkEjM1NTQwNz
g0Ny2MDAyOTQyOHIPcmftdgFwHdyxRl
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 "946A0A1AC8245696B9C6A6F35942690B"
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.

#### 21.1.6 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


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.

21.2 Temporary authorized access

Introduction of STS

OSS can temporarily perform authorized access through the Alibaba Cloud STS (Security Token Service). Alibaba Cloud STS is a web service that provides a temporary access token to a cloud computing user. Using STS, you can grant access credentials to a third-party application or federated user (you can manage the user IDs) with customized permissions and validity periods. Third-party applications or federated users can use these access credentials to directly call the Alibaba Cloud product APIs or use the SDKs provided by Alibaba Cloud products to access the cloud product APIs.
• You do not need to expose your long-term key (AccessKey) to a third-party application and only need to generate an access token and send the access token to the third-party application.
• You can customize the access permission and validity of this token.
• You do not need to care about permission revocation issues. The access credential automatically becomes invalid when it expires.

Using an app as an example, the interaction process is shown as follows:

1. Log on as the app user.

   App user IDs are managed by the customer. Customers can customize the ID management system, or use an external web account or OpenID. For each valid app user, the AppServer can precisely define the minimum access permission.

2. The AppServer requests a security token (SecurityToken) from the STS.

   Before calling STS, the AppServer needs to determine the minimum access permission (described in policy syntax) of the app user and the expiration time of the authorization. Then the security token is obtained by calling the STS’ AssumeRole interface.

3. The STS returns a valid access credential to the AppServer, including a security token, a temporary AccessKey (AccessKeyID and AccessKeySecret), and the expiry time.

4. The AppServer returns the access credential to the ClientApp.

   The ClientApp can cache this credential. When the credential becomes invalid, the ClientApp needs to request a new valid access credential from the AppServer. For example, if the access credential is valid for one hour, the ClientApp can request the AppServer to update the access credential every 30 minutes.
5. The ClientApp uses the access credential cached locally to request Alibaba Cloud Service APIs. The cloud services perceives the STS access credential, and relies on STS to verify the credential and correctly respond to the user request.

For more information on the STS Security token, refer to role management in the RAM usage guide.

Call AssumeRole of the STS interface to obtain valid access credentials. You can also directly use STS SDK to call this method.

Use STS credentials to construct signed requests

After obtaining the STS temporary credential, the client of the user creates a signature using the security token (SecurityToken) and temporary AccessKey (AccessKeyId and AccessKeySecret) in the credential. The method for constructing an authorized access signature is basically the same as using the AccessKey of a root account to add a signature to a header. Pay attention to the following two points:

- The signature key used by the user is the temporary AccessKey (AccessKeyId and AccessKeySecret) provided by the STS.
- The user needs to carry the security token (security token) in the request header or in the URI as a request parameter. These two manners are alternative. If both manners are selected, OSS returns an InvalidArgument error.

  - The header x-oss-security-token: SecurityToken is carried in a request header. When CanonicalizedOSSHeaders of the signature is calculated, x-oss-security-token is taken into consideration.
  - Parameter security-token=SecurityToken is carried in the URL. When CanonicalizedResource of the signature is calculated, security-token is taken into consideration and considered as a sub-resource.

21.3 CDN-based OSS acceleration

Background

Structure of traditional products without static-dynamic separation (however, performance encounters a bottleneck as traffic increases).
Product structure implementing static-dynamic separation (a flexible structure supports massive user traffic).
Scenarios

- Massive access to static files, high server loads, and I/O problems, causing slow user access.
- Large volumes of static files and insufficient storage space.
- Massive access to static files distributed across various regions.
- Fast and concurrent download of mobile update packages in large volumes within a certain time period.

Structural description

As the storage source for massive file volumes, OSS stores static images, video files, downloaded packages, app update packages, and other resources. As OSS is the origin site for CDN, OSS files can be obtained through CDN accelerated delivery by accessing nearby CDN nodes.

Structural advantages:

- Reduces the load on Web servers, and directs the access to all static files to CDN.
- Provides the lowest storage fees. OSS storage fee is only half that of ECS disks.
- Provides massive storage capacity, without the need to consider structural upgrade.
- Minimizes traffic fees. Apart from a small amount of additional origin retrieval traffic, the majority of the traffic is CDN traffic. And it’s cost is lesser than the Internet traffic for direct access from OSS.

Case study

A common website is used as an example. A recently established website www.acar.com, is an automotive news and discussion website which is built on PHP. The main site stores 10 GB of image resources and some JS files. An ECS instance is purchased to store all program codes and MySQL database is installed on the ECS instance.

As access traffic continues to grow, many users report that the website access speed experiences a slow down such as loading of pictures and website response consumes time. The website’s technical staff notices that users are uploading an increasing number of images and the total size will soon exceed 1 TB.

The technical staff can use OSS and CDN to optimize the website structure to achieve static-dynamic separation shown in the preceding figure. It enhances user experience, and keep their costs at a manageable level.
The specific solution and procedures are as follows:

1. Sort out the website program code on the ECS instance by storing dynamic programs and static resources in different directories for better management.
   - Create a directory named **Images** for storing the website’s high-definition images.
   - Create a directory named **Javascript** for storing all JS scripts.
   - Create a directory named **Attachment** for storing all images and attachments uploaded by users.

2. Create a bucket.
   
   Select the bucket region based on your ECS region and select the permission option Public Read. You have to make sure that the bucket name corresponds to one of the directories created on the ECS instance, for example, acar-image-bucket. For more information, see *Create a bucket*.

3. Enter image.acar.com as the domain for the HD videos and images on your website. For more information, see *Manage a domain name*.

4. Upload files to verify the CDN effect.
   
   a. Upload all image files in the Images directory created in Step 1 on the ECS instance to acar-image-bucket. For more information, see *Upload objects*. You can use an OSS client to complete the upload process conveniently.
   
   b. Get the CDN address for this file. The address format is your CDN domain +"/"+file name. For more information, see *Get object URL*.
   
   c. Upload image files one by one.

5. Repeat the preceding steps to upload the files in the other two directories, and create the CDN-based OSS buckets acar-js-bucket and acar-csimages-bucket.

6. In the ECS system, find the access code for the static files and replace the access URL with the CDN domain. Users access static files on your website in OSS+CDN mode without occupying your ECS resources.

**Note:**

If you want to automatically synchronize user-uploaded files to acar-csimages-bucket, see the OSS SDKs and the PutObject section of the API documentation. This provides information on how to perform automatic upload at the code level.
CDN automatic refresh

If you use Alibaba Cloud CDN with a bound CDN domain that points back to an OSS source, you can use OSS’s CDN cache automatic refresh function. OSS automatically refreshes CDN when the data is overwritten (for example, when an existing file is overwritten or deleted). A origin retrieval operation is performed to obtain the overwritten file from OSS, so you do not need to explicitly call the CDN refresh interface. The URL refresh rules are as follow:

CDN domain + / + Object

For example, if the uploaded file test.jpg is overwritten in the bucket bound to the CDN domain image.acar.com, OSS refreshes the image.acar.com/test.jpg URL. The time required by the refreshed URL to take effect is determined by CDN’s guaranteed refresh time, which is typically less than 10 minutes.

To activate CDN-based OSS acceleration, enable the Refresh CDN cache function on the bucket Domain Management page.